Dissertation
submitted to the
Combined Faculties for the Natural Sciences and for Mathematics
of the Ruperto-Carola University of Heidelberg, Germany
for the degree of
Doctor of Natural Sciences

presented by

Diplom-Physiker Markus Merschmeyer
born in Ibbenbüren

Oral examination: 10th November, 2004

Production and Flow of Neutral Strange Particles in Ni+Ni Collisions at 1.93 AGeV

Referees: Prof. Dr. Norbert Herrmann
Prof. Dr. Johanna Stachel

Die vorliegende Arbeit präsentiert Ergebnisse einer Untersuchung der Produktion und des Flusses von K^0 und Λ^0 in Ni+Ni-Kollisionen bei einer Energie von 1.93 AGeV. Hierzu wurden mehr als 10^8 zentrale Ereignisse ausgewertet, die mit einer verbesserten Datenaufnahme innerhalb von 12 Tagen aufgezeichnet worden waren. Dies war nur mit Hilfe eines schnellen Datenreduktionsalgorithmus möglich, der im Rahmen dieser Arbeit auf einem digitalen Signalprozessor implementiert wurde. Die Verteilungen von Steigungsparametern und Produktionswahrscheinlichkeiten des K^0 entsprechen denen des K^+, die schon früher untersucht worden waren. Vergleiche mit Transportmodell-Rechnungen (UrQMD, IQMD) zeigen, daß keines der Modelle in der Lage ist, diese Verteilungen von K^0 und Λ^0 konsistent zu beschreiben. Es konnte gezeigt werden, daß der integrale und der differentielle Fluß des K^0 mit denen des K^+ übereinstimmen, was bestehende Hinweise auf ein repulsives K^+N-Potential unterstützt. Ein Vergleich zwischen Λ^0 und Protonen zeigt keine Unterschiede im integralen oder differentiellen Fluß. Daher stellt sich die Frage nach der Stärke eines In-Medium-ΛN-Potentials und seiner Auswirkung auf die Größe des beobachteten Flusses. Desweiteren wurde eine Methode zur Rekonstruktion von Ξ^-- und $\Sigma^-(1385)$-Baryonen entwickelt, mit welcher Kandidaten beider Teilchen identifiziert werden konnten. Die geringe Rekonstruktionseffizienz macht allerdings eine Bestätigung durch ein Experiment mit wesentlich größerer Statistik notwendig.

This thesis presents results on the production and the flow of K^0 and Λ^0 in Ni+Ni collisions at an energy of 1.93 AGeV. The analyzed data sample of 10^8 central events has been recorded during 12 days using an upgraded data acquisition with a fast online data reduction algorithm. Part of this work was the implementation of this algorithm on a digital signal processor. The slope parameter and yield distributions of the K^0 are found to agree with those of previously measured K^+. Comparisons to transport model calculations (UrQMD, IQMD) show that none of the models is able to consistently describe those distributions for both, K^0 and Λ^0. The integral and the differential directed flow of the K^0 are found to agree with that of the K^+, supporting the evidence for a repulsive K^+N potential. Comparing the directed flow of Λ^0 and protons, no significant difference is found between both for the integral as well as the differential flow. This brings up the issue of the strength of the in-medium ΛN potential and its relation to the magnitude of the observed flow. A method to reconstruct Ξ^- and $\Sigma^-(1385)$ baryons has been developed and candidates of both particles have been identified. The very small reconstruction efficiency requires confirmation through an experiment with a significantly larger statistics, though.

Contents

List of Figures

List of Tables

Overview

This thesis presents results on the production of strange particles in relativistic heavy-ion collisions at a beam energy of 1.93 AGeV in the Ni+Ni system. The data were measured in 2003 with the FOPI detector at the SIS[1] accelerator of the 'Gesellschaft für Schwerionenforschung' (GSI) in Darmstadt.

The field of heavy-ion physics is introduced briefly and, taking into account the present knowledge of strangeness production, open questions are summarized. The motivation for further studies of hadronic matter properties at finite densities by measurements of neutral- and multi-strange particles is given. Afterwards, the setup of the FOPI detector is described with an emphasis on the Central Drift Chamber which was used to reconstruct the strange particles. The third chapter describes the new data acquisition system which was used in this experiment. Part of this work was the implementation of a fast algorithm for the online reduction of the drift chamber data on a digital signal processor. Thus, it was possible to collect a high-statistics data sample of 10^8 events within 12 days of beam time.
Technical details of the data analysis and the reconstruction of strange particles are explained in chapters four and five, respectively. The determination of an efficiency correction for the measured particle yields by employing a simulation of the detector is described in chapter six. Due to the nearly complete coverage of the phase space in the experiment the efficiency correction introduces no bias and the estimation of an error for quantitative results is given.
Chapter seven presents the physics results of this work. For the first time it is possible to study production and collective behaviour of neutral strange particles covering nearly the full phase space. Measured yields, phase space distributions and collective effects of strange particles are compared to other experimental findings and to model predictions in chapter eight. New insights from comparisons of Λ and proton emission patterns are reported.
The status of the search for multi-strange particles and short-lived strange resonances is briefly described in chapter nine. Finally, the results are summarized and an outlook is given.

---

[1] Schwerionensynchrotron

Chapter 1

Introduction

This work presents an investigation of strange particle production in heavy-ion collisions. These collisions are used as a tool to create a system of hot and dense nuclear matter. Within this scenario, new particles are produced which – in addition to the common *up*- and *down* quarks – can contain *strange* quarks or even heavier quark flavours, if the collision energy is sufficiently high. The strange particles are proposed as probes for the change of hadronic properties inside the compressed nuclear matter. At SIS energies, kaons, ϕ mesons, Λ^0 and Σ hyperons can be produced in heavy-ion collisions.

The following chapter briefly introduces the field of heavy-ion physics. First, nuclear matter properties are described and the concepts of an equation of state and in-medium effects are recalled. Afterwards, the study of nuclear matter properties by means of heavy-ion collisions is explained. Existing results on strange particle production are summarized and the proposal for an experiment measuring Ξ^- production is motivated.

1.1 Nuclear Matter

Studying the properties of nuclear matter has attracted a lot of interest for both, experiment and theory, in the past decades [1, 2]. Key questions are the existence of new phases, the possibility of describing bulk nuclear matter by an equation of state and the presence of in-medium effects which manifest in the change of hadron properties.

1.1.1 Phases of Nuclear Matter

Nuclear matter can appear in different phases, depending on temperature and density. Some of them are already established experimentally, others

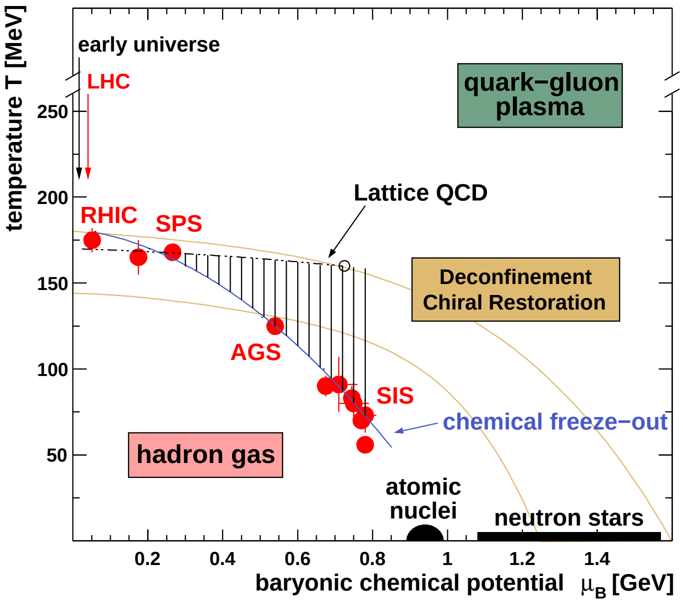


Figure 1.1: Phase diagram of nuclear matter. Depending on temperature and baryonic chemical potential the nuclear matter system can be a hadron gas, a quark-gluon plasma or in some other phase. The red points mark the chemical freeze-out curve which was determined using a statistical model [3]. The picture was taken from this reference and modified.

are still being searched for. Figure 1.1 sketches the phase diagram of nuclear matter depending on the temperature T and the baryonic chemical potential μ_B which is equivalent to the density ρ.

Normal nuclei are commonly described by the droplet model assuming that they behave like a liquid. This phase exists at temperatures lower than 10 MeV and densities close to that of normal nuclear matter, ρ_0. A solid phase could exist in neutron stars where temperatures are low but the density is significantly higher.

The gaseous phase, a so-called *hadron gas*, exists at temperatures below 100 MeV and densities smaller than ρ_0. This dilute system of hadrons is the outcome of nuclear collisions at relativistic energies. Starting from the ratios of produced particles it is possible to determine points of chemical freeze-out in the T vs. μ_B plane within the framework of a statistical model assuming equilibrium. The red points in figure 1.1 denote results for various collision energies reached at different accelerators [4, 5, 6, 7, 8] thus defining a chemical freeze-out curve.

At very high densities and/or very high temperatures, quarks are no longer confined inside hadrons but are predicted to form a *quark-gluon plasma* [9]. In the passage from a hadron gas or a normal nucleus to a deconfined state, a phase transition may occur. This also opens the possibility of the existence of a critical point in the phase diagram. The nature of the transition and the question whether it coincides with a restoration of chiral symmetry [10, 11] are under discussion.

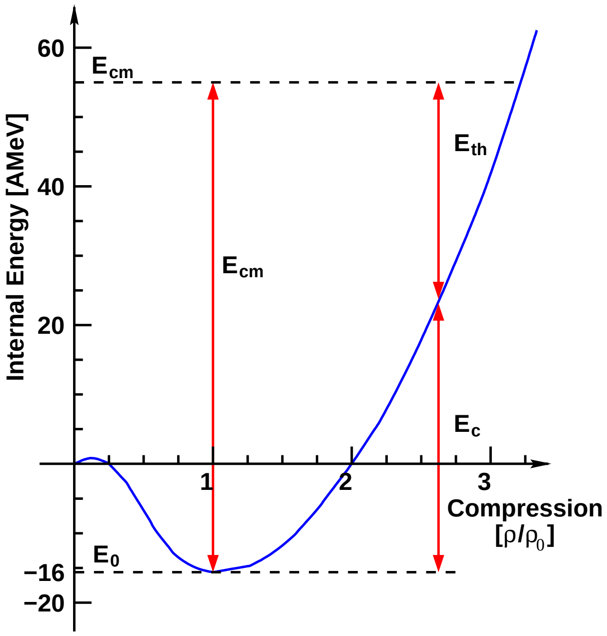


Figure 1.2: Schematic illustration of the equation of state at T=0. The ground state energy is E_0= -16 MeV per nucleon and $\rho = \rho_0$. In a collision with a center-of-mass energy E_{cm} the curve describes the division of that energy into thermal (E_{th}) and compressional energy (E_c).

1.1.2 Equation of State

If an equilibrium is reached during a heavy-ion collision, the system can be described by an equation of state (EOS) [12], i.e. a relation between three thermodynamical variables. For nuclear matter, these are for example the internal energy E, the temperature T and the density ρ. The internal energy can be decomposed into a thermal (E_{th}) and a compressional part (E_c)

$$E(\rho, T) = E_{th}(\rho, T) + E_c(\rho, T = 0) + E_0 \ , \tag{1.1}$$

where the compressional part is defined at zero temperature and E_0 is the ground state energy. An illustration of the density dependence of the internal energy per nucleon at T=0 is given in figure 1.2.

An infinite system of nuclear matter at zero temperature and zero pressure has a saturation density ρ_0= 0.17 nucleons/fm^{-3}. Due to the absence of surface effects the binding energy at this density is $E_0 \approx$ -16 MeV per nucleon compared to approximately -8 MeV for an average nucleus. If nuclear matter is compressed, e.g. during a heavy-ion collision, the center-of-mass energy E_{cm} of the collision is divided into compressional energy E_c and thermal energy E_{th}. The thermal energy itself is used for thermal excitation and particle production. The amount of energy which is needed to compress the system is determined by the *incompressibility* K_∞:

$$K_\infty = 9\rho_0^2 \left[\frac{d^2 E_c}{d\rho^2}\right]_{\rho=\rho_0} \ . \tag{1.2}$$

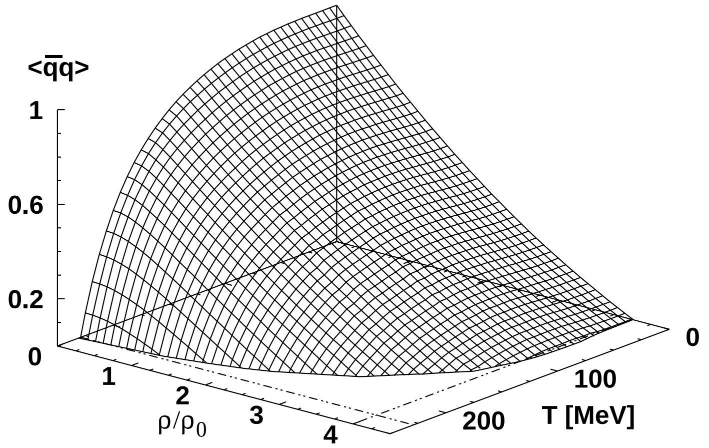


Figure 1.3: Dependence of the quark condensate $|\langle q\overline{q}\rangle|$ on temperature T and nuclear matter density ρ in the framework of the Nambu-Jona-Lasinio model [18].

For an incompressibility of approximately 200 MeV, the equation of state is called 'soft'. At higher values of K_∞ ($\approx$ 400 MeV), the EOS is called 'stiff'

The experimental determination of the incompressibility can be done e.g. by astrophysical observations, by investigations of giant monopole resonances or by heavy-ion collisions. The sub-threshold production of K^+ in those collisions was proposed as a probe for the EOS [13]. However, the experimental determination of the incompressibility by studying K^+ production is rendered difficult due to the possible presence of in-medium effects (described in the following subsection). An investigation of the system-size dependence of K^+ production has found evidence for a soft EOS [14, 15].

1.1.3 In-Medium Effects

In hot and dense nuclear matter, chiral symmetry can be partially restored. Due to the non-zero value of the quark condensate $|\langle q\overline{q}\rangle|$, this symmetry is explicitly broken and the effective quark masses are of the order of several hundred MeV. Calculations according to the Nambu-Jona-Lasinio model [16, 17] predict a decreasing value of the quark condensate when temperature and/or density of nuclear matter are increased [18]. A graphical illustration of this result is given in figure 1.3.

In the case of a (partially) restored chiral symmetry the effective masses of the quarks are reduced to their current masses which are of the order of some MeV for u,d and about 150 MeV for the s quark. Therefore, the production of quark-anti-quark pairs is energetically much easier. A changing value of the quark condensate also indirectly affects hadron properties. The Gell-Mann, Oakes and Renner relation [19] connects quark condensate and quark current masses (m_i^0) to hadron masses and hadron decay constants.

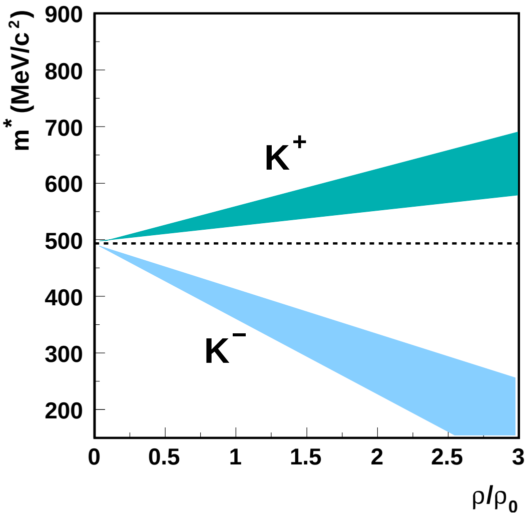


Figure 1.4: Effective mass of K^+ and K^- as a function of nuclear matter density. While the K^+ mass is slowly growing with increasing density, the K^- mass strongly decreases. This trend is predicted by several theoretical calculations [26, 27, 28, 29] for kaons at rest.

For the kaons one expects for example

$$m_K^2 f_K^2 = -\frac{1}{2} \left(m_u^0 + m_s^0 \right) \langle u\overline{u} + s\overline{s} \rangle + O(m_s^2) \ , \tag{1.3}$$

where m_K is the kaon mass and f_K the kaon decay constant. Similar effects have been predicted for the vector mesons whose effective masses in the medium can be described by a scaling law [20]. A mass change will affect production and propagation of particles produced in hot and dense nuclear matter. Therefore, experimental investigations of the particle yield and collective effects can deliver information on the nature of a change of hadronic properties.

The consequences of (partial) chiral symmetry restoration are subsumed under the concept of *in-medium effects* (IME). Various experimental probes for these effects have been proposed. Vector mesons decaying into dilepton pairs are one example because the properties of their decay products are not affected by the hadronic environment [20, 21, 22, 23, 24, 25]. A second example are strange particles, in particular the K^+. The K^+N elastic scattering cross section is of the order of 10 mb which corresponds to a mean free path of several fm. Due to their strange quark content K^+ mesons are less probable to be reabsorbed in nuclear matter as compared to the K^- (due to baryonic resonances and strangeness exchange) or non-strange hadrons. In the SIS energy regime, strange particles mostly are produced below the production threshold where in-medium effects will have a most pronounced effect on the particle production yields.

The effect of in-medium modifications is illustrated by the trend of theoretical predictions for the density dependence of effective kaon masses [26, 27, 28, 29], plotted in figure 1.4. For the effective mass of the K^+, a slight increase is expected as the density of the surrounding nuclear matter increases while the K^- mass exhibits a more pronounced drop. A comparison of calculations concerning kaon IME is given in [30].

The question, whether hadronic properties are influenced by the surrounding nuclear medium, has far reaching consequences in astrophysics. The condensation of K^- in dense nuclear matter [31, 32] can soften the EOS of nuclear matter which, in a supernova explosion, would favour the formation of a black hole instead of a neutron star [33, 34, 35, 36].

In order to incorporate in-medium effects into transport model descriptions of heavy-ion collisions, potentials are used which act on the effective masses of the particles. While a repulsive KN potential is employed to describe the effects for the K^+, the K^- effects are modeled with an attractive KN potential.

1.2 Heavy-Ion Collisions

Heavy-ion collisions allow the investigation of hot and dense nuclear matter in the laboratory. For a long time, experimental investigations have been performed at different accelerator facilities providing ion beams with energies ranging from a few MeV up to 200 GeV. Fixed target experiments were (and are) performed at the BEVALAC, the SIS, the AGS and the SPS accelerators providing maximum beam energies of approximately 2 AGeV, 12 AGev and 160 AGeV, respectively. Recently, the RHIC started its operation, colliding two ion beams with energies of 100 AGeV each. Future experiments are prepared at the LHC which is currently built at CERN. This collider will be able to accelerate ions up to energies of 2.7 ATeV.

The physics questions addressed at the different accelerators concern various aspects of nuclear matter properties, e.g. the degree of equilibrium reached in the collisions, the equation of state and the compressibility of nuclear matter, the presence of in-medium effects and the search for new phases of nuclear matter. Answers to those questions will have impact on astrophysics because nuclear matter properties determine the dynamics of supernova explosions, the properties of neutron starts or the creation of matter shortly after the big bang.

When two ions collide, the reaction geometry can be described by the vector of the projectile velocity and the impact parameter b which define the *reaction plane*. An illustration of a nucleus-nucleus collision is given in figure 1.5. At beam energies of 2 AGeV the projectile approaches the target

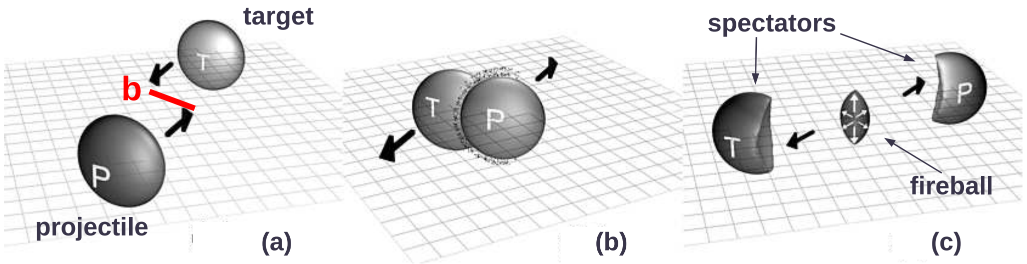


Figure 1.5: Geometrical representation of a heavy-ion collision. The projectile nucleus (P) hits the target nucleus (T) with an impact parameter b. In the overlap region, a fireball of excited nuclear matter is created. The reaction plane is indicated by the grid. The picture was taken from [37].

nucleus with a velocity of about 90 % of light velocity (a). In the overlap region of projectile and target nucleus (b), a fireball of hot and dense nuclear matter is created [38] which expands and disassembles into a hadron gas. Meanwhile, the spectator fragments, containing the nucleons which did not participate in the reaction, leave the collision zone (c) [39]. The typical timescale of collision at SIS energies is about 5–10 fm/c which corresponds to approximately $2{\cdot}10^{-23}$ s.

The centrality of the collision is determined by the magnitude of the impact parameter. Head-on collisions with an impact parameter close to zero are called central collisions. In this case the highest densities are reached and the participating nucleons undergo many collisions. Therefore, this scenario is interesting for studying strange particle production. Further topics are the degree of stopping and equilibration reached in the reaction [40, 41].

Collisions with larger impact parameters are called mid-central or peripheral. Here, collective effects, the so-called *flow* of particles, are investigated [42]. While particle emission is azimuthally symmetric in central events, different patterns are observed in peripheral collisions. Directed flow denotes the emission of particles into the reaction plane which occurs when the spectator fragments are deflected by the expanding fireball (cf. picture '(c)' in figure 1.5). Squeeze-out is observed perpendicular to the reaction plane when the emission of particles into the reaction plane is blocked by the spectators. In order to investigate flow effects, the reaction plane of an event has to be determined from quantities which are accessible experimentally. The method applied in this work is described in chapter 4.5.

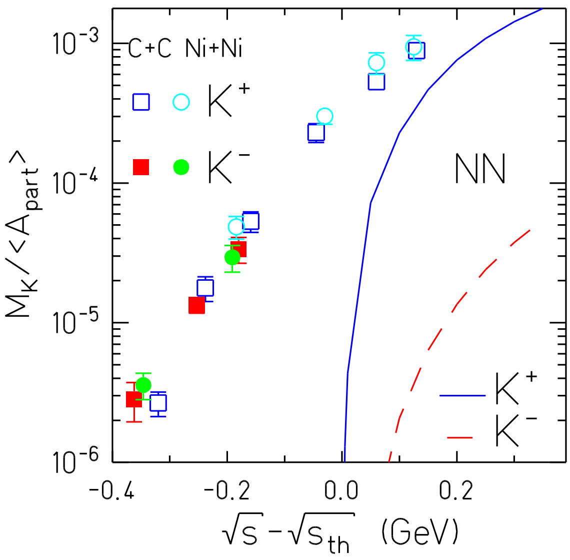


Figure 1.6: Subthreshold production of K^+ and K^- in C+C and Ni+Ni reactions measured by the KaoS experiment [46, 47, 48]. The solid and the dashed lines denote the yields for nucleon-nucleon collisions.

1.3 Measurements of Strange Particles

Strange particles are unique probes for investigating nuclear matter properties in heavy-ion collisions [43, 44, 45]. The strong interaction conserves strangeness, therefore an s quark is always produced in association with an $\bar{s}$ antiquark. The reabsorption of a strange particle in the fireball would require another particle containing the proper (anti-)quark for the annihilation. At SIS, this is not very likely to happen because strange particles are rare in this energy regime.

The strange particles which were produced in a collision decay weakly at a timescale of about 10^{-10} s which is approximately twelve orders of magnitude larger than the timescale for the heavy-ion reaction. By then the particles are well outside the collision zone and their decay products cannot be obscured by hadronic interactions. At SIS energies, strangeness production generally happens below or close to the free nucleon-nucleon threshold. The creation of K^+ (K^0) and Λ^0 (Σ^0) via the process NN $\rightarrow$ $K\Lambda$N needs an energy of more than 1.6 AGeV in the free reaction. The associated production of K^+ and K^- (NN $\rightarrow$ K^+K^-NN) requires a minimum energy of 2.5 AGeV. In heavy-ion reactions, nucleons will undergo more than one collision and higher resonances can be produced. If these resonances participate in a another collision, the production thresholds for strange particles are lower compared to the free NN thresholds.

The production of K^- and K^+ was studied by the KaoS collaboration [47, 48] in C+C and Ni+Ni collisions. The results are compiled in figure

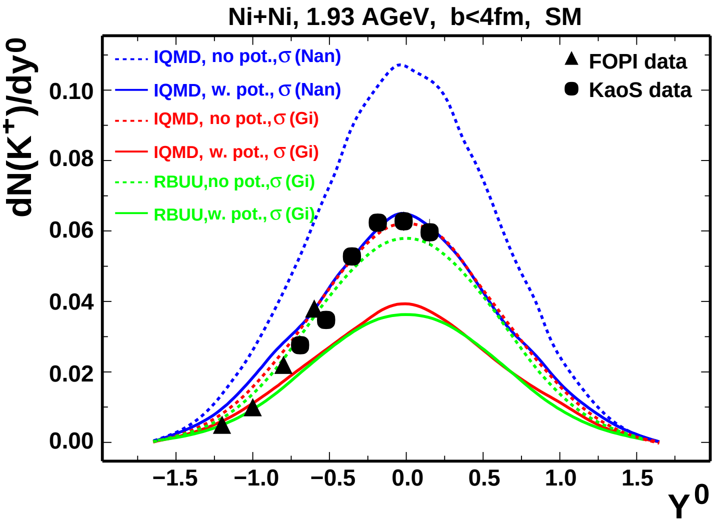


Figure 1.7: Differential yield distribution of K^+ mesons measured by the KaoS [49] and FOPI [50] experiments. IQMD and RBUU model predictions with and without in-medium effects are indicated by the solid and the dashed lines, respectively. The picture was taken from [53].

1.6. In contrast to nucleon-nucleon collisions where the production is possible only above the threshold (indicated by the solid and dashed lines in the plot), here both particles are already produced far below the threshold. Furthermore,the production cross sections for K^- and K^+ were found to agree when comparing at the same excess energy above the nucleon-nucleon threshold. The results can be interpreted in terms of in-medium effects for the K^- causing a drop of the effective mass although it remains unclear to which extent multi-step processes and strangeness-exchange reactions contribute to the measured effects.

The differential yield distribution of K^+ mesons was measured by the KaoS and FOPI experiments in Ni+Ni reactions [49, 50] at 1.93 AGeV. A comparison of the experimental results to IQMD [51] and RBUU [52] model predictions with (solid lines) and without (dashed lines) in-medium effects is presented in figure 1.7 (an explanation of the models is given in appendix A). The data show a good agreement between both experiments. Comparing to theoretical predictions, no definite conclusion about the presence of in-medium effects can be drawn. This is a consequence of model-dependent assumptions on the cross sections. Several of them, e.g. those concerning Δ resonances, are experimentally inaccessible and have to be parameterized or estimated from other known cross sections. When using the same set of cross sections, both models agree in their predictions (cf. red and green lines in the plot) [53].

More evidence for in-medium effects comes from measurements of the K^+ differential flow in Ni+Ni collisions at 1.93 AGeV depicted in the left plot of figure 1.8. An anti-correlation between the flow of K^+ and protons was observed for low momentum particles [54]. Comparisons to RBUU model

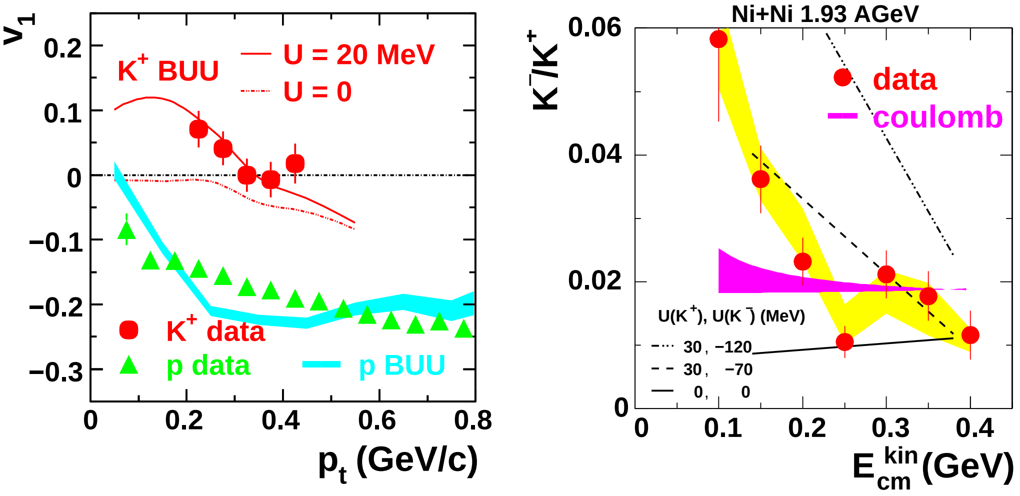


Figure 1.8: Differential directed flow of the K^+ [54] (left) and kinetic energy dependence of the K^-/K^+ ratio [55] (right) measured in Ni+Ni collisions by the FOPI experiment.

predictions show that, using a repulsive in-medium potential of 20 MeV, the results are described well. However, an uncertainty remains due to the influence of the Coulomb repulsion between proton and K^+.

Measurements of the K^-/K^+ ratio in heavy-ion collisions are favourable because on the one hand many systematic effects cancel and on the other hand the ratio is predicted to be more sensitive to in-medium effects. Studies concerning this ratio were done by the FOPI and KaoS experiments [55, 56]. Plotted as function of the kinetic energy E_{cm}^{kin} in the center-of-mass system (figure 1.8, right plot) the ratio drops with increasing energy. Transport model (RBUU) calculations (given by the black lines) can reproduce this effect using a repulsive potential of 30 MeV for the K^+ and an attractive potential of -70 MeV for the K^-, respectively. The contribution of the Coulomb force (indicated by the pink area) was estimated and found to be too small to account for the observed effect. Investigating the centrality dependence of the K^-/K^+ ratio in Ni+Ni and Au+Au Collisions at 1.5 AGeV it was found that the ratio is nearly constant as a function of the number of participants (cf. figure 1.9). QMD calculations (hatched area) fail to describe the ratio, while a statistical model predicts a constant ratio of about 0.02 using $\mu_b = 770$ Mev and $T = 63$ MeV [57, 58]. This result is taken as a hint that the K^- and K^+ production mechanisms are correlated by strangeness exchange reactions.

Strangeness production at higher energies (AGS, SPS, RHIC) can be sen-

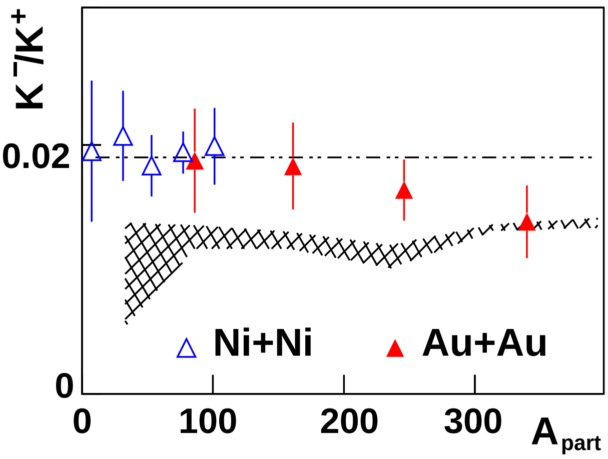


Figure 1.9: Dependence of the K^-/K^+ ratio on the number of participants, A_{part}, measured in Ni+Ni and Au+Au collisions by the KaoS experiment [56]. The line denotes the prediction of a statistical model and the hatched area indicates the result of a QMD model calculation.

sitive to the presence of a quark-gluon plasma. In this phase chiral symmetry is restored and the production of $s\bar{s}$ pairs is energetically more favourable. Therefore, an enhancement of strange particle production should indicate the formation of a QGP in heavy-ion collisions at these energies. In this context, multi-strange baryons like Ξ^- or Ω^- are of particular interest.

Measurements of Ξ^- production at the AGS were performed by the E810 and E895 collaborations [59, 60]. A Ξ^- yield of 0.25±0.04 per event and a Ξ^-/Λ ratio of 0.12±0.02 were found in Si+Pb collisions at 14.6 AGeV. This result was more than a factor four above cascade model predictions.

The near-threshold Ξ^- production in 6 AGeV Au+Au collisions was studied later on in semi-central collisions ($b < 7$ fm). For the most central collisions ($b \approx 0$ fm) a $\Xi^-/(\Lambda^0+\Sigma^0)$ ratio of 0.017±0.005 was reported (cf. figure 1.10, left plot). The particle yield is in agreement with statistical and transport model predictions which imply that already at these collision energies the multi-strange hadron production approaches chemical equilibrium.

At SPS, hyperon production has been measured in Pb+Pb collisions and proton-induced reactions by the NA57 experiment [61]. The Λ and Ξ^- yields per wounded nucleon relative to p+Be as a function of the collision centrality are depicted in the right plot of figure 1.10 for a beam energy of 158 AGeV. An enhancement of strangeness production increasing with the strangeness content of the hyperons is observed.

The excitation function of the $\Xi^-/(\Lambda^0 + \Sigma^0)$ ratio is plotted in figure 1.11. Results from AGS, SPS and RHIC experiments can qualitatively be described by a statistical model (indicated by the dashed line). According to this model, also at SIS energies a finite value of this ratio is observable.

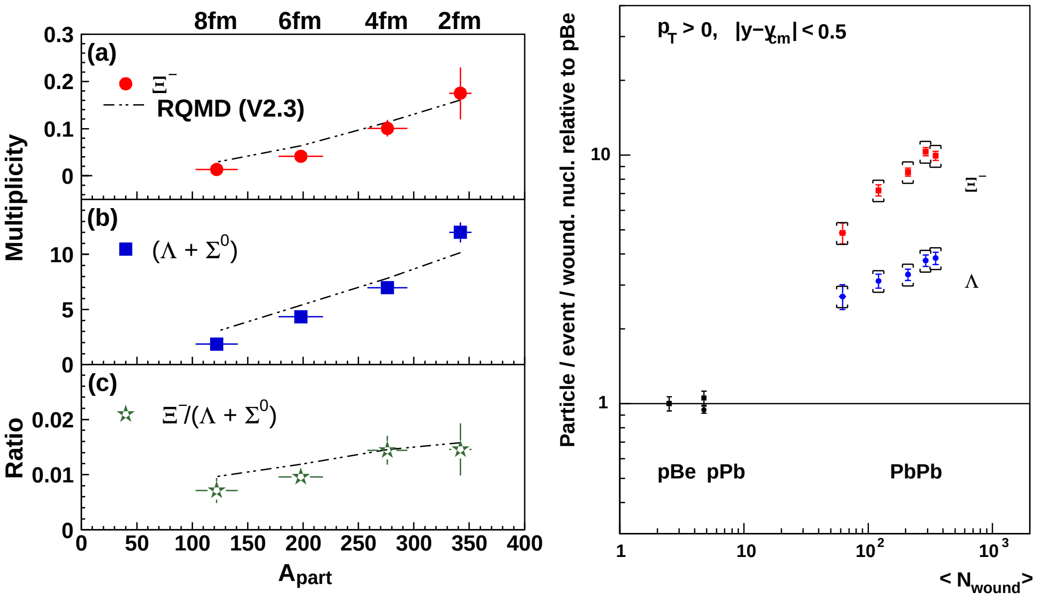


Figure 1.10: Results of measurements of Λ and Ξ^- production by the E895 collaboration [60] at AGS (left) and by the NA57 collaboration [61] at SPS (right).

Consequences

To shed more light on the question of in-medium effects and to support already existing results, further steps have to be done:

- A cross-check of the existing yield and flow results for the K^+ can be done by measuring the K^0 which is not influenced by the Coulomb force.

- The question whether the K^- yield is linked to that of the Λ^0 via the $\pi\Lambda^0 \rightarrow K^-N$ channel can be studied by a measurement of the Λ^0 production in combination with model calculations.

- Comparisons of model calculations to experimental data concerning charged kaons have produced first hints to in-medium effects. In order to get a consistent picture more particle species have to be incorporated into the comparisons. Measurements of K^0 and Λ^0 production will provide more information.

- The enhanced production of multi-strange baryons observed at higher energies raises the speculations about the nature of the enhancement mechanism. An investigation of deep-subthreshold Ξ^- production at SIS energies would contribute to the understanding of this mechanism in a purely hadronic scenario.

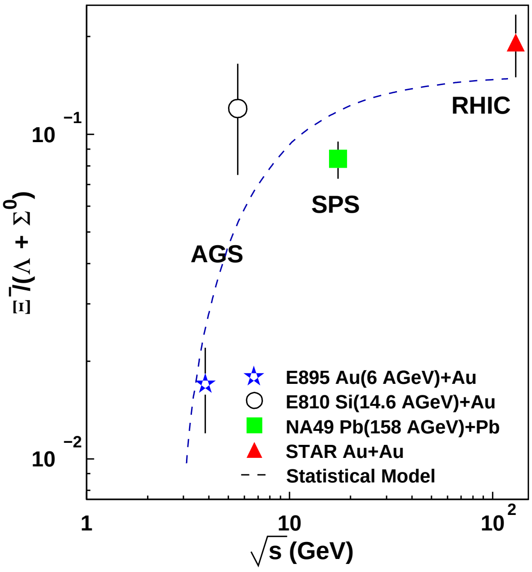


Figure 1.11: Excitation function of the $\Xi^-/(\Lambda^0 + \Sigma^0)$ ratio. The dashed blue line indicates the prediction of a statistical model. The picture was taken from [60].

Experiments to tackle the points given above can be done with the FOPI detector. Statistics of the order of 10^8 events and a large phase space coverage will be needed to get conclusive answers.

1.4 The Ξ^- Experiment

In 2001, a high-statistics experiment was proposed which aimed at investigating the Ξ^- production in Ni+Ni collisions at beam energies of approximately 2 AGeV. Extensive simulations were performed to find the highest possible geometrical acceptance of the detector. An optimum was found for a target position lying 40 cm in front of the nominal position.

Using a statistical model prediction of the Ξ^-/K^+ ratio and the previously measured K^+ yield, a production probability of $(8.5\pm2.0)\cdot10^{-5}$ per central collision is determined. This implied that about $1.2\cdot10^8$ central events needed to be recorded in order to find traces of the Ξ^-. As a side effect, these data can also be analyzed for neutral strange particles like K^0 and Λ^0.

The main challenge of this experiment was to collect the huge number of events within a beam time of about two weeks. This required a smooth running of the detector and a high-speed data acquisition. Therefore, a part of this work is devoted to the implementation of an algorithm for online reduction of drift chamber data.

The proposal was accepted and as a consequence, the so-called 'S261' experiment was performed in January and February of 2003.

Chapter 2

The FOPI Experiment

The FOPI detector is located at the 'Schwerionensynchrotron' (SIS) of the 'Gesellschaft für Schwerionenforschung' (GSI) in Darmstadt, Germany. This facility provides ion beams with kinetic energies from 0.1 to 2 AGeV. Recently, also secondary proton and pion beams became available. FOPI is a 4π detector which means that it is capable of measuring the products of heavy-ion reactions in the full solid angle. It was designed for detecting charged particles, i.e. light mesons (π, K), protons, deuterons, tritons and heavier fragments.

Since its assembly by an international collaboration in 1990, the experimental setup has been changed and extended several times [62, 63]. Physics topics were the study of nuclear fragmentation [64, 65], pion production [66, 67, 68], the production of strange particles (Kaons, Λ, ϕ) [54, 55, 69] and investigations of flow [70, 71].

The following sections describe the FOPI detector and the experiment which delivered the data analyzed in this work. The 'Central Drift Chamber' (CDC), the sub-detector which was used to reconstruct strange particles, is explained in more detail.

2.1 Setup of the Detector

The FOPI detector comprises several sub-detectors [72]. A schematic drawing of the detector is presented in figure 2.1. The following sub-detectors are used in the experimental setup:

- **Start- and Halo Detectors:** The start detector consists of a thin scintillator foil tilted by 45° around the vertical axis. It is placed about 2 m in front of the target. Light produced by ions passing through the foil is detected by two photomultipliers perpendicular to the beam.

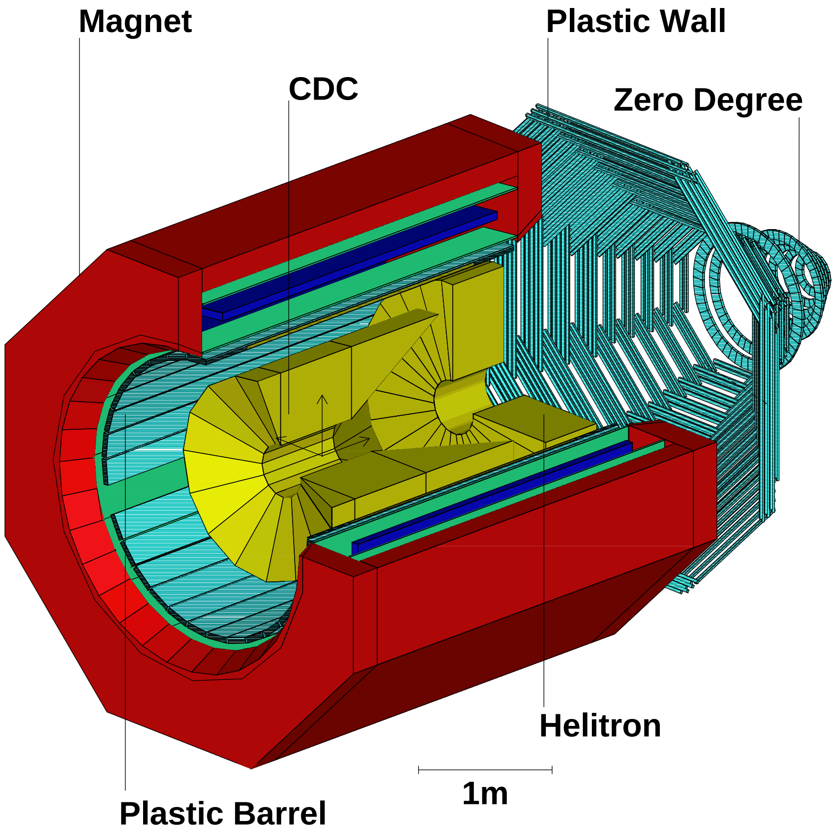


Figure 2.1: Schematic drawing of the FOPI detector.

This detector is used to count incoming ions and to define the reference time t_0 (start time) for all other detectors.

The beam quality is monitored by two halo detectors 200 cm and 10 cm in front of the target. Four scintillator bars read out by photomultipliers are arranged in a cross shape around the beam. Thus a rectangular area of adjustable size is defined. The halo detectors deliver a veto signal whenever an ion passes them.

- **Magnet:** The superconducting magnet produces a homogeneous solenoidal field parallel to the beam axis. Its dimensions are 3.3 m in length and 2.4 m in diameter. The Barrel, the Helitron and the CDC are placed inside the magnet. The nominal field of 0.6 T is reached at a current of 720 A. Charged particles passing through the field move along helical trajectories. Their transverse and total momenta can be extracted from the curvature in the x-y plane and the polar angle ϑ.

- **Central Drift Chamber (CDC):** The CDC is the detector which delivered the data analyzed in this work. Therefore, it will be explained in more detail in section 2.2.

- **Helitron:** The Helitron is a radial drift chamber which covers the full azimuthal range and polar angles from 10° to 22°. It is filled with a gas mixture of 88 % Ar, 10 % iso-C_4H_{10} and 2 % CH_4 and is divided into 24 sectors separated by cathode wires generating the drift field. The middle of each sector contains a plane of 54 potential wires and 53 sense wires extending radially from the inner radius to the outer radius of the chamber. The potential wires produce the electric field for the electron multiplication and the signals are measured on the sense wires. The Helitron yields information about momenta and energy loss of charged particles.

- **Barrel:** The inner wall of the magnet is covered by the Barrel time-of-flight (TOF) detector. It is built out of 180 scintillator bars with dimensions of $150\times4\times3$ cm^3 grouped in 30 modules with 6 bars each. The Barrel provides information about the energy loss, time-of-flight and position. It covers polar angles from 27° to 57° and nearly the complete azimuthal range.

- **Plastic Wall (PLAWA):** The TOF detector in the forward direction is the Plastic Wall covering the ϑ range from 7° to 25°. 512 bars of plastic scintillators, read out at both ends, are arranged in 8 sectors with 64 bars each. The active length of the bars rises from 45 cm at the inner radius to 165 cm at the outer radius of the detector. The Plastic Wall measures energy loss ($\Delta E \approx \sqrt{E_1 \cdot E_2}$) and time-of-flight ($\approx (t_1 + t_2)/2$). In addition it yields information about the position along a scintillator bar ($x \propto (t_1 - t_2)/2$).

- **Zero-Degree Counter (ZDC):** This detector is built out of seven concentric rings, each consisting of 36 trapezoidally-shaped plastic scintillators read out on one side. It covers the polar angular range from 1° to 7° and is used for the detection of particles in the forward direction. Like the Plastic Wall, it measures energy loss and time-of-flight of the particles.

2.1.1 The S261 Experiment

The experiment which delivered the data for this work was called the S261 experiment and was performed in the beginning of 2003. The SIS produced a beam of $^{58}_{28}\mathrm{Ni}^{28+}$ at a kinetic energy of 1.93 AGeV. The average beam intensity was 4×10^{6} ions per spill at a length of 10-12 s and a duty cycle of about 80 %. The target was a foil of enriched (95 %) $^{58}_{28}\mathrm{Ni}$ with a size of 15×15 mm^2 and a thickness of 405 μm. This corresponds to an areal density of 356 mg/cm^2.

For a sufficiently thin target, the interaction probability can be calculated with the following formula

$$p_{react} = \rho x \cdot \frac{N_A}{A} \cdot \sigma_{tot}^{geom} = \rho x \cdot \frac{N_A}{A} \cdot \left(\sqrt[3]{A_p} + \sqrt[3]{A_t}\right) r_0 \tag{2.1}$$

where ρ is the density of the target material, x the target thickness, N_A the Avogadro constant, A the atomic weight of the target material and σ_{tot}^{geom} the total geometrical cross section for a reaction between projectile and target nucleus. The total geometrical cross section can be expressed by the atomic number of projectile and target nucleus, A_p and A_t, and a radius r_0 ($\approx$ 1.2 fm). Using the values given above an interaction probability of 1.5 % is calculated.

The target position in this experiment was shifted by -40 cm along the beam (z) axis with respect to its position in previous experiments. This was done in order to increase the CDC's acceptance for the Ξ^- baryon. In addition, the target which is located approximately 60 cm behind the exit window of the beam pipe was placed inside a helium-filled gas bag. This reduces the number of background events which originate from collisions between gas molecules and ions from the beam in the area around the target.

2.2 The Central Drift Chamber

This section will provide a short introduction into the detector's technical features, the physics effects which lead to the generation of detectable signals and on how to reconstruct and identify particles from those signals. More information on the physics of the detector can be found in books like [73, 74]. Further details about the CDC are provided in [75, 76].

2.2.1 Technical Details

The CDC is a jet chamber. This type of drift chamber was first used in the JADE experiment at PETRA (DESY) [77] and later on in the OPAL experiment at LEP (CERN) [78, 79]. Due to the short drift paths, electron drift times are of the order of a few μs which allows to operate the chamber at high event rates. In addition, the resolution is not limited by diffusion so that events with a high local track density can be measured. Therefore this

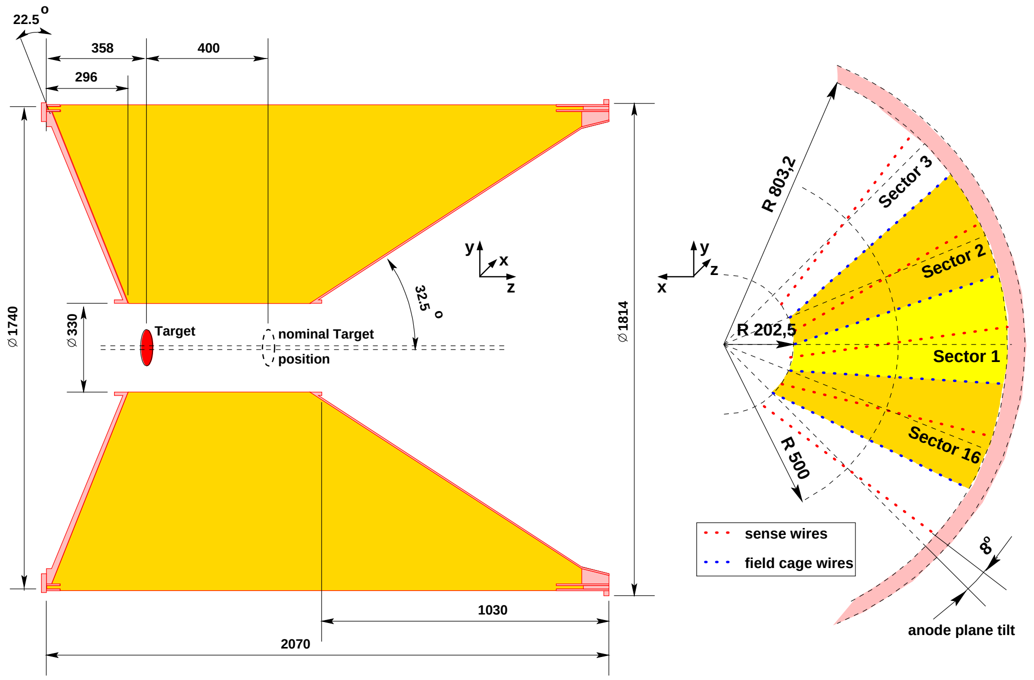


Figure 2.2: Longitudinal (left) and transversal (right) cross section drawings of the CDC. The target is shifted by -40 cm with respect to its nominal position.

chamber can be used in heavy-ion physics where the overall track density is high and good double-track resolution is needed.

The chamber is placed inside the magnet in a volume where the magnetic field is homogeneous to a precision of better than 1 %. It has a cylindrical shape with conical end caps at the front and the back side. This corresponds to an acceptance for polar angles ranging from 23° to 113° while the azimuthal coverage is complete (see figure 2.2).

The CDC is azimuthally divided into 16 sectors. The active volume of a sector is surrounded by a field cage which separates it from the neighboring sectors and limits the inner and outer radius at 20 cm and 80 cm, respectively. The field cage of a sector consists of 252 wires ($\oslash$ 125 μm) parallel to the beam direction (see figure 2.3, top left picture). The anode plane in the middle of each sector contains 61 potential wires ($\oslash$ 125 μm) and 60 sense wires ($\oslash$ 20 μm) placed alternately at distances of 0.5 cm. This plane is not aligned radially but is tilted by 8° in order to reduce the problems with mirror tracks (see below). In addition the sense wires are displaced alternately by $\pm$ 200 μm (staggering) with respect to the anode plane (see figure 2.3, top right picture).

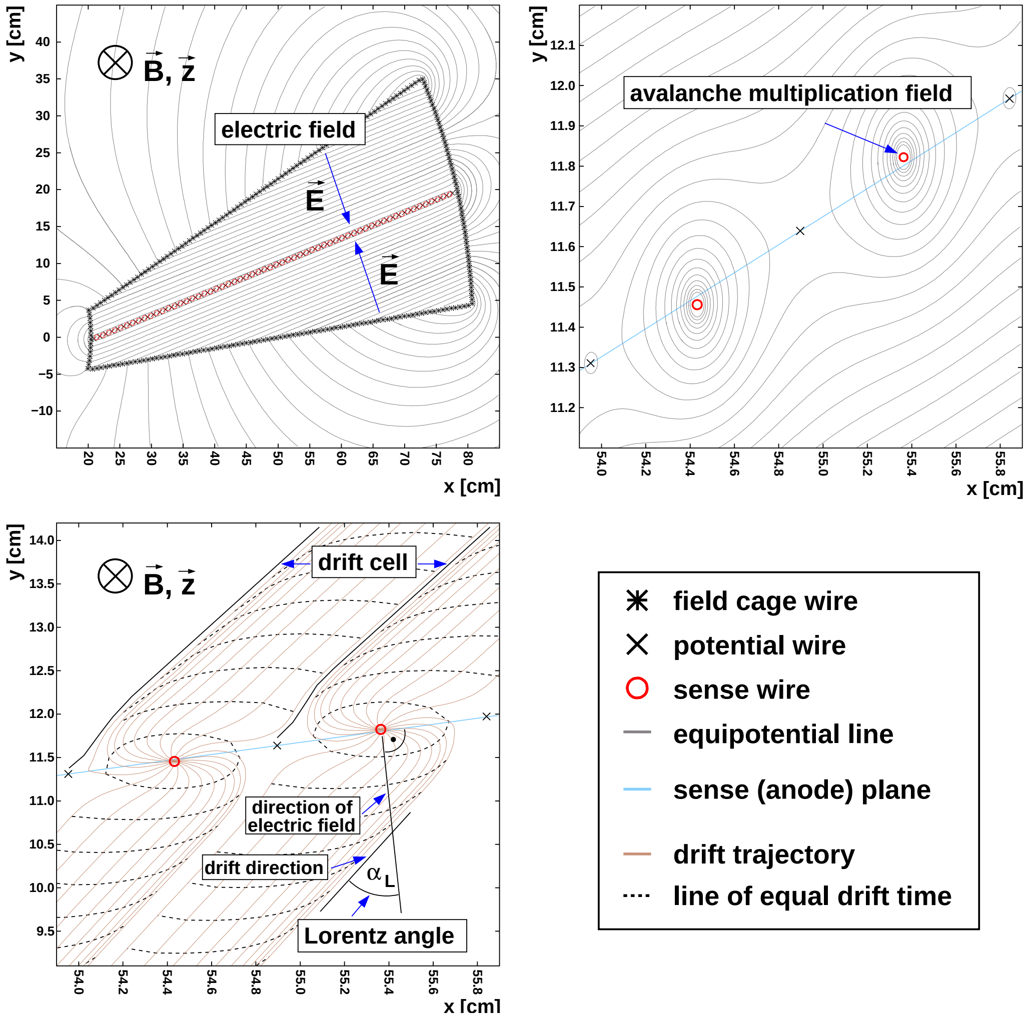


Figure 2.3: Illustration of the electric field (upper left plot), the avalanche multiplication field (upper right plot) and the drift trajectories of electrons (lower plot) in a CDC sector. The pictures were generated by the GARFIELD program [80] which is used for the simulation of gas-filled detectors.

The electric drift field is generated by the wires of the field cage (see figure 2.3, top left picture). The nominal drift voltage of -15 kV which is distributed to the wires by a voltage divider chain produces a homogeneous drift field of about 800 V/cm . The electric field for the electron multiplication (see figure 2.3, top right picture) is generated by applying a voltage of -1275 V to the potential wires while the sense wires are set to ground (zero) potential.

The detector is filled with a gas mixture of Ar, iso-C_4H_{10} and CH_4 at a ratio of 88:2:10. The isobutane and methane gases act as quenchers which allow to operate the detector at higher gas gains with an extended region of good proportionality.

2.2.2 Signal Detection

When a charged particle (e.g. p, $\pi^{\pm}$) traverses the gas-filled volume of the CDC it loses energy by ionizing atoms or molecules of the gas. The mean energy loss is described by the Bethe-Bloch formula (see appendix B). Once created electrons and ions are separated by the electric field. The positive ions are accelerated towards the field cage while the electrons are accelerated towards the sense wires in the anode plane. On their way both, electrons and ions, undergo collisions with atoms or molecules of the gas. However, this does not lead to the production of more ion pairs in the electric field far from the anode plane because the kinetic energy the particles gain between two successive collisions is not high enough. Therefore, the particles acquire an average velocity v_D, the drift velocity. This velocity has the same magnitude as the mean velocity v of particles at thermal energies which are described by a Maxwell distribution

$$v = \sqrt{\frac{8kT}{\pi m}} \tag{2.2}$$

At room temperature v is of the order of a few cm/μs for electrons and about a factor 100 lower for ions. In the case of the CDC the drift velocity is approximately 4.5 cm/μs.

The presence of the magnetic field which is perpendicular to the electric field influences the direction of the drift velocity. This is illustrated in the lower picture of figure 2.3. Each drift cell is no longer parallel to the electric field but tilted by the Lorentz angle α_L. A drift cell is some region within the active volume of a sector assigned to a specific sense wire. It is defined by the fact that all electrons created by ionization in that drift cell will end up on the sense wire it belongs to. This can be seen from the drift trajectories in the picture. The Lorentz angle also depends on the gas mixture, the pressure as well as the electric and magnetic field. For the CDC it is about 42° to 45°. The longest drift path in this configuration is about 22 cm which corresponds to a maximum drift time of $t_{D,max} \approx 5$ μs.

After this time, the drifting electrons reach the vicinity of the sense wires. Within a radius of 0.5 cm around those wires the electric field gets strongly inhomogeneous (see figure 2.3, top right picture). Thus the electrons which - due to their low mass - are easily accelerated, gain sufficient kinetic energy to produce more ion pairs. Electrons created in these secondary processes themselves can produce more ion pairs and so on. This results in the formation of an avalanche. The charge Q which is finally collected on a sense wire is orders of magnitude larger than the charge $n_0 \cdot e$ created by the primary ionization process (M : multiplication factor):

$$Q = n_0 \cdot e \cdot M \tag{2.3}$$

Due to the operating conditions of the detector and without distortions by space charge effects Q is still proportional to the primary charge. For the given gas mixture and the field configuration around the sense wire M is of the order of 10^4 to 10^5. An avalanche detected on a sense wire by measuring its charge is called *hit*.

2.2.3 Hit Reconstruction

To get information on charged particles produced in a collision it is necessary to reconstruct their tracks in the CDC. The first step is to determine the positions of the primary ionizations which caused the hits that were detected. In the x-y plane they can be found by extrapolating back from the known wire position taking into account Lorentz angle, drift time and drift velocity. The z position along the wire can be calculated with the method of charge division [81]. A detailed description of the hit reconstruction procedure is given in [76, 82]. The position resolution in x-y is better than 500 μm, the z coordinate can be determined with a precision of 5 - 7 cm. Due to the design of the chamber the electrons created by ionization in each sector can drift towards the sense plane from two directions (see top left picture of figure 2.3). This induces an ambiguity for the hit position in the x-y plane because there are two possible positions, one on either side of the sense plane.

2.2.4 Tracking and Calibration

Once all hit positions are determined, the track finding procedure is started. This is done with the help of complex algorithms which are explained e.g. in [76, 83]. There are different classes of algorithms called 'global' and 'local' depending on whether the algorithm is acting on all hits or on a subset of hits at a time. For tracking the S261 data a local algorithm called 'Local 2' was used. The different position resolutions in x-y and z require a special tracking strategy. First the tracking algorithm is searching for hits which lie on an arc in the 2-dimensional transverse plane. When this is found, the z coordinates of the hits are used to determine the inclination angle ϑ of the track.

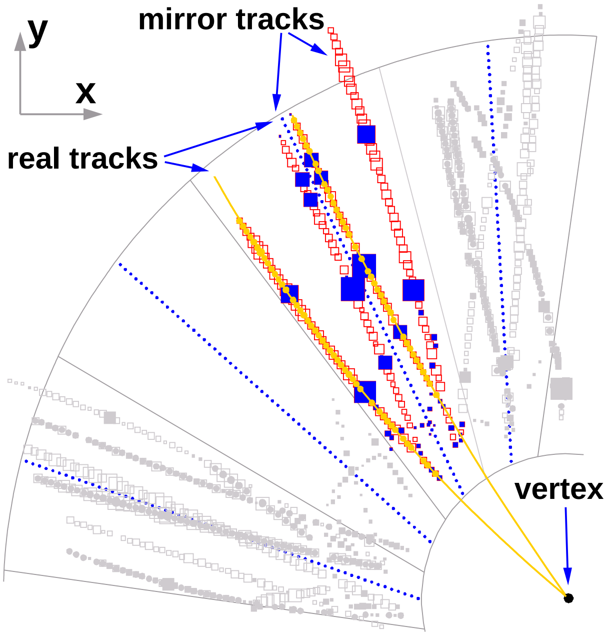


Figure 2.4: Illustration of tracks and their 'mirror images' in a CDC sector. The real tracks are marked by the yellow arcs which originate from the primary vertex. The corresponding mirror tracks (hollow red squares) on the opposite side of the sense plane do not point to the primary vertex and may have hits outside the active volume of the CDC.

Due to the ambiguity of the x-y position of the hits one has to ensure that the track search ignores so-called mirror tracks. This is plotted in figure 2.4. The technical features explained in section 2.2.1 help to reduce this problem: As a result of the tilting of the sense plane mirror tracks never originate from the event vertex but miss it by a distance of several centimeters. The staggering of the sense wires leads to a better fit quality for the real track in comparison to the mirror track. Furthermore, as a result of the Lorentz angle, mirror tracks include hits with positions outside the active volume of the CDC.

After tracking and excluding the mirror tracks the three parameters v_d, α_L and t_0 have to be calibrated (t_0 is the 'total time offset' which is needed to calculate the drift time t_D). This is done through selecting tracks which either cross a sector border (v_D, α_L) or the sense plane (t_0, α_L) and adjusting the parameters so that there are no discontinuities in the track left. To get a good calibration the whole process of hit reconstruction, tracking and calibrating is repeated several times always using the set of parameters determined in the step before. The result of this procedure for one event in the CDC is illustrated in figure 2.5.

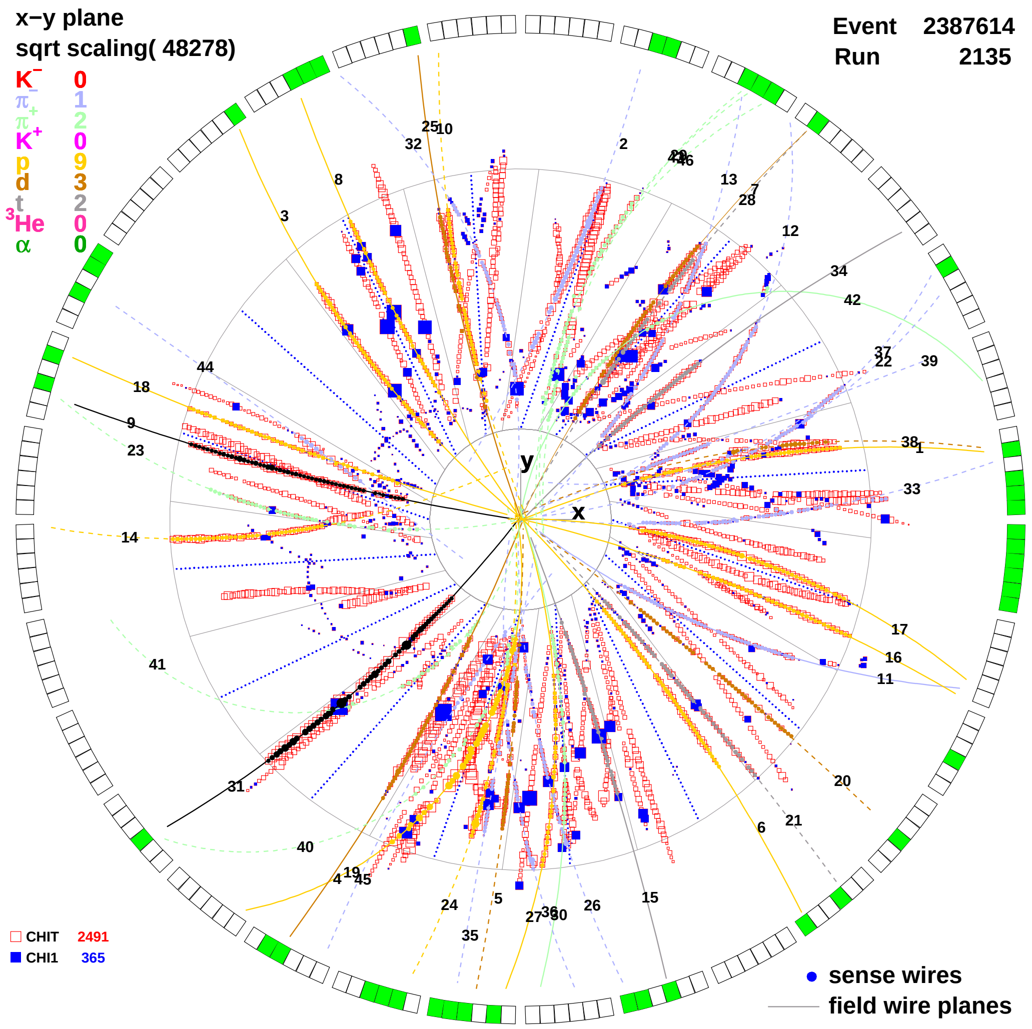


Figure 2.5: Illustration of a central Ni+Ni collision measured with the CDC in the transverse plane. The blue dotted lines indicate the sense planes, and the red hollow squares mark the positions of reconstructed hits. Tracks which were found and accepted by the tracking routine are labelled by the filled squares and the colored arcs.

2.3 The Trigger

In a fixed-target experiment at SIS energies, event rates of the order of a kHz are easily reached. A trigger system is used to filter out the interesting events, for example those where strangeness is most likely to be produced. Thus the amount of data to be stored is significantly reduced and background events are suppressed.

The incoming ions are counted by the start detector and the halo detectors monitor the alignment and the focusing of the beam. If no veto signals were produced by the halo detectors, a possible event can be accepted by the trigger. Different classes of events can be defined through requests to the beam quality, the number of charged particles detected in the forward Plastic Wall and the information of other detectors, e.g. the CDC or the Barrel. The charged particle multiplicity in the Plastic Wall is a measure of the centrality of a heavy-ion collision because as the impact parameter decreases the charged particle multiplicity is expected to rise. The trigger information coming from start detector and Plastic Wall is available after approximately 500 ns.

To enrich events of a certain type in relation to other events, *down-scale factors* are applied meaning that e.g. every central event is accepted while only every n$^{\mathrm{th}}$ mid-central and every m$^{\mathrm{th}}$ peripheral event is being taken. The experimental cross section can then be calculated as follows:

$$\sigma_{\mathrm{exp.,trig.}} = \frac{1}{N_{\mathrm{beam}} n_{\mathrm{targ.}}} \cdot \frac{N_{\mathrm{trig.}}}{ds_{\mathrm{tot.}} ds_{\mathrm{trig.}}} \tag{2.4}$$

and

$$ds_{\mathrm{tot.}} = \frac{N_{\overline{\mathrm{inh.}}}}{N_{\mathrm{raw}}} \ , \quad ds_{\mathrm{trig.}} = \frac{N_{\mathrm{acc.}}}{N_{\overline{\mathrm{inh.}}}}$$

with

$\sigma_{\mathrm{exp.,trig.}}$: experimental cross section for a given trigger type,
N_{beam} : number of incoming ions per second,
$n_{\mathrm{targ.}}$: number of target particles per cm^2,
$N_{\mathrm{trig.}}$: number of observed events for a given trigger per second,
$ds_{\mathrm{tot.}}$: dead-time induced down-scale factor,
$ds_{\mathrm{trig.}}$: hardware down-scale factor,

$N_{\overline{\mathrm{inh.}}}$: number of events outside dead-time of detector/DAQ,
N_{raw} : total number of events,
$N_{\mathrm{acc.}}$: number of events accepted.

To determine the triggered cross section the numbers (N_{beam}, $N_{\mathrm{trig.}}$, N_{raw} and $N_{\overline{\mathrm{inh.}}}$) are counted event-wise by LeCroy 4434 32-fold 32 bit CAMAC scalers.

Chapter 3

The Data Acquisition

The task of the data acquisition (DAQ) is to reliably collect, preprocess, combine and store the data from all sub-detectors. The preprocessing step is needed to reduce the huge amount of data coming from the CDC and the Helitron. Each of those detectors on average produces 2 MB of raw data per event and together they account for 80 - 90 % of all data. For a high statistics experiment with event rates of at least 100 Hz this would mean a data rate of more than 500 MB per second. With the given network and computing infrastructure, it is impossible to transfer and store the experimental data at this rate.

The chapter first describes the current data acquisition system and then explains the SAM3 module, a hardware component which is needed to reduce the drift chamber data. The module uses a digital signal processor (DSP) on which a special data reduction algorithm is executed. After briefly introducing the DSP, the working principle of the data reduction algorithm and its implementation on the SAM3 module will be explained. Finally, the performance of the DAQ system in the 2003 experiment will be reported.

3.1 The Data Acquisition System

Recording more than $1{\cdot}10^8$ central collisions within a beam time of two weeks at a duty cycle of 80 % and a spill length of 10 s requires an event rate of larger than 100 Hz. Therefore, the old FOPI DAQ system (described e.g. in [84]) had to be upgraded. The upgrade particularly concentrated on the readout electronics of the drift chambers. Due to the drift times of about 5 μs they are the slowest detectors, thus determining the overall speed of the full DAQ system. Several parts of the old drift chamber readout system (explained e.g. in [76]) which was designed similar to that of the OPAL experiment at

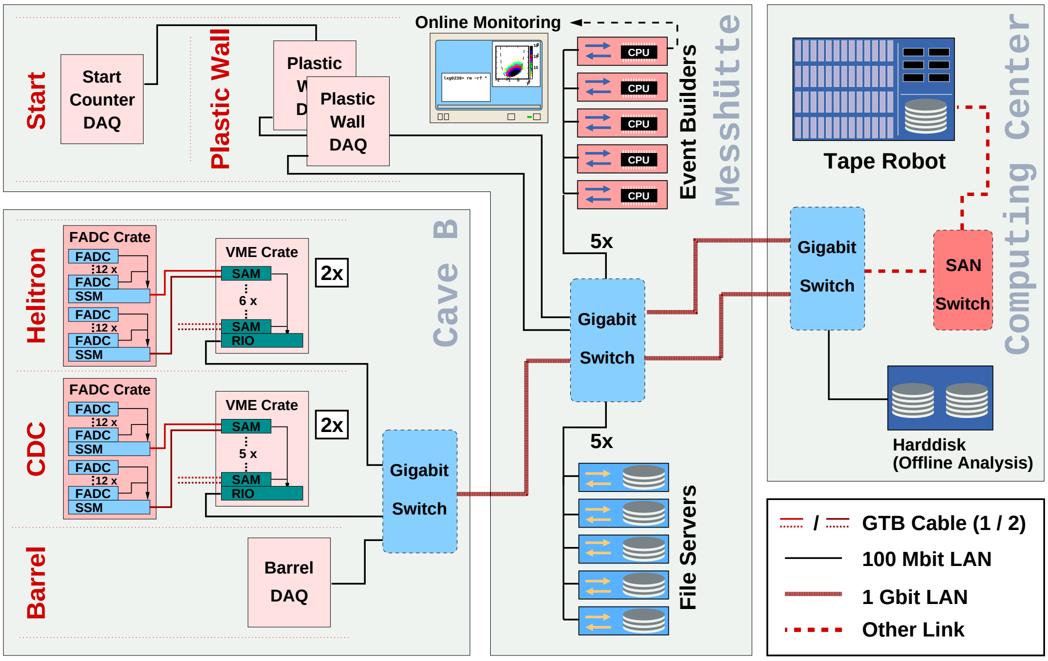


Figure 3.1: Schematic drawing of the DAQ setup in the S261 experiment.

CERN [78, 79], were replaced with newly developed components. The current acquisition system is sketched in figure 3.1 and includes the following parts:

- **Flash Analog-to-Digital Converters (FADC):** After a hit on a wire has been picked up by the preamplifiers the resulting signal pulse has to be digitized for further processing. FADCs with a sampling frequency of 100 MHz are used to provide a time resolution of 10 ns with a dynamic range of 8 bit. During the sampling the digitized values are stored in a ring buffer with a depth of 1024 samples. This corresponds to a maximum time range of 10.24 μs. With drift velocities of 4 - 5 cm/μs and drift paths not longer than 25 cm a time range of 500 - 600 μs would already be sufficient to digitize all hits.

  The dynamic range of the FADC is effectively extended to 10 bit due to the nonlinear characteristics of its internal preamplifier. It has the form
  $$S' = \frac{256 \cdot S}{256 + a \cdot S} \ , \quad S = c \cdot U \tag{3.1}$$
  with the output signal height S' ($0 \leq S' \leq 255$), the input signal height S, the compression factor a (≈ 0.68), the input voltage U (≤ 600 mV) and the conversion factor c ($\approx 255/600$ mV). The extended dynamical

range is desirable because the energy loss of both, minimum-ionizing particles (MIPs) like pions and heavy fragments, can then be measured very accurately.

Each FADC module has eight independent inputs which means that it can process the data from four wires read out on both ends. 24 modules are arranged in two groups in a so-called crate which provides the cooling system, the power supply and the data bus for the readout. Each group of FADC modules is assigned to one SSM module (see below). Altogether 240 FADCs (in ten crates) are needed for the CDC readout, 288 FADCs (in twelve crates) for the Helitron readout.

- **Scanning and Sampling Module (SSM):** When a valid trigger signal is received the sampling of the FADCs is stopped and their data is collected by the SSMs. Each module reads and combines the data from 12 different FADCs. In total, 44 modules are used in the CDC and Helitron data acquisition. A first step of reducing the data is performed at this stage: The application of a certain threshold discards sample values which are smaller. Depending on the number of hits and the noise on each wire a data reduction of about a factor 5 - 10 can be achieved.

  To be able to adjust the module's functions to changing requests later on the core functions were implemented on an FPGA[1] with a VHDL[2] program. Depending on how good the program can be realized on the FPGA (placing of the logical blocks and routing of the connections between them) the clock frequency of the SSM can be chosen between 20 - 40 MHz.

  The combined FADC data is transferred from the SSM to the SAM via a so-called GTB[3] cable. This connection offers a 16 bit wide bus with a frequency of 10 MHz. Thus up to data rates of 20 MB/s can be achieved. The SSM offers two different modes for the data transmission. In the (slow) *address mode* the master (SAM) provides an arbitrary address within the SSM's hit-buffer memory and receives the data word stored at this address. This mode is used for calibrating the FADC pedestals and initializing the SSM at the startup of the DAQ. The *token mode* exploits the full bus speed and is used when transferring zero-suppressed FADC data from the SSM to the SAM during

---

[1] Field-Programmable Gate Array

[2] VHSIC (Very High Speed Integrated Circuits) Hardware Description Language

[3] Teräte Transport Bus

the experiment. Here the SSM directly writes the entire content of its hit-buffer memory into the GTBRAM on the SAM.

- **"Steuerungs- und Auslese-Modul" (SAM):** The second, more complex step of online data reduction is done by the SAM. Each of the 22 modules in the drift chamber DAQ receives the data collected by two SSMs via the GTB cables. The data is then processed by two DSPs[4] and stored in an external memory for further collecting. The function of the SAM module is explained in more detail in the following section.
- **RIO3:** After the data reduction by the SAM modules is finished, the remaining data is collected by the RIO3 [85] and sent to the event builders. This module is a commercially available Power-PC real-time computer. It is also used for the calibration of the FADC modules (via SAM and SSM) and the initialization of the SSM (via SAM) at the start of the data acquisition. Furthermore it loads the data reduction code (see appendix D) into the DSP memory and starts the processor after initializing the other modules.
- **Event builders, File servers and Tape Robot:** The "event builders" are used to combine the data delivered by all sub-detectors into full events. Each event builder is assigned to one of five file servers providing a disk space of 200 GB. The file servers are used for buffering the raw data files from the experiment before they were written to a tape robot in the computing center of GSI. This device provides space of more than 100 TB in LTO2 cartridges with a capacity of 100 GB per volume. More than 5000 raw data files (around 1 GB per file) were written during the beam time.

3.2 The SAM Module

The SAM module [86], developed by J. Hofmann at GSI, is the hardware component needed for the online data reduction. In addition to FOPI it is also used in other experiments (e.g. HADES [87]) and the hadron cancer therapy [88]. Figure 3.2 shows a schematic drawing of the 3rd version of the SAM module (SAM3). The DSPs, the different memories, the devices for communication via GTB and VME[5], the FPGAs for the control of the module's operation are depicted in this representation. Furthermore the

---

[4] Digital Signal Processor(s)

[5] Versa Module Europa

paths for addresses, data and control commands between those devices are included.

Each DSP has direct access to the three different kinds of RAM[6]: The 16 MB large SDRAM[7] (unused in the present FOPI setup) and the two 128 kB dual-ported[8] RAMs called GTBRAM and VRAM. The GTBRAM receives the data collected by the SSM when reading out the FADCs.

The GTBRAM is organized in 64k blocks with 16 bits of memory per block and can be accessed by the SSM and the DSP. From the GTBRAM the data is then moved into the internal RAM of the DSP or directly into the VRAM, depending on the mode of operation. This memory is organized in 32k blocks with 32 bits of memory each. It can also be accessed by the DSP and the RIO3. The data transfer between GTBRAM, DSP internal RAM and VRAM is controlled by the "DSPLOG" FPGA and the external memory interface (EMIF) of the DSP.

Finally the RIO3 can retrieve the data from the VRAM by a so-called D64 block transfer via the VME bus and send them to the event builders via a 100 Mbit LAN[9]. In order to increase the speed the transfer is organized in 64 bit long data words compared to the 32 bit transfers normally employed. The "VMELOG" FPGA is responsible for the communication between RIO3 and SSM via the host-port interface (HPI) of the DSP and for the programming of the other FPGAs.

3.3 The C6711 Digital Signal Processor

The DSP used on the SAM3 module is the Texas Instruments TMS320C6711 [89, 90]. This processor is performing fixed-point and floating-point operations at a clock speed of 100 MHz corresponding to a cycle time of about 10 ns. A diagram of the functional blocks, the memories and the DSP core is given in figure 3.3.

The core part of the DSP contains 8 independent functional units, $2\cdot16$ 32-bit registers, the instruction fetch, dispatch and decode units, control registers and further logic for testing, emulation and interrupt handling. The functional units are divided into two groups each containing four units. Both groups are connected to one of the two data paths of the CPU[10] (A and

---

[6] Random Access Memory

[7] Synchronous Dynamic RAM

[8] Each memory block can be independently accessed by two hosts at the same time.

[9] Local Area Network

[10] Central Processing Unit

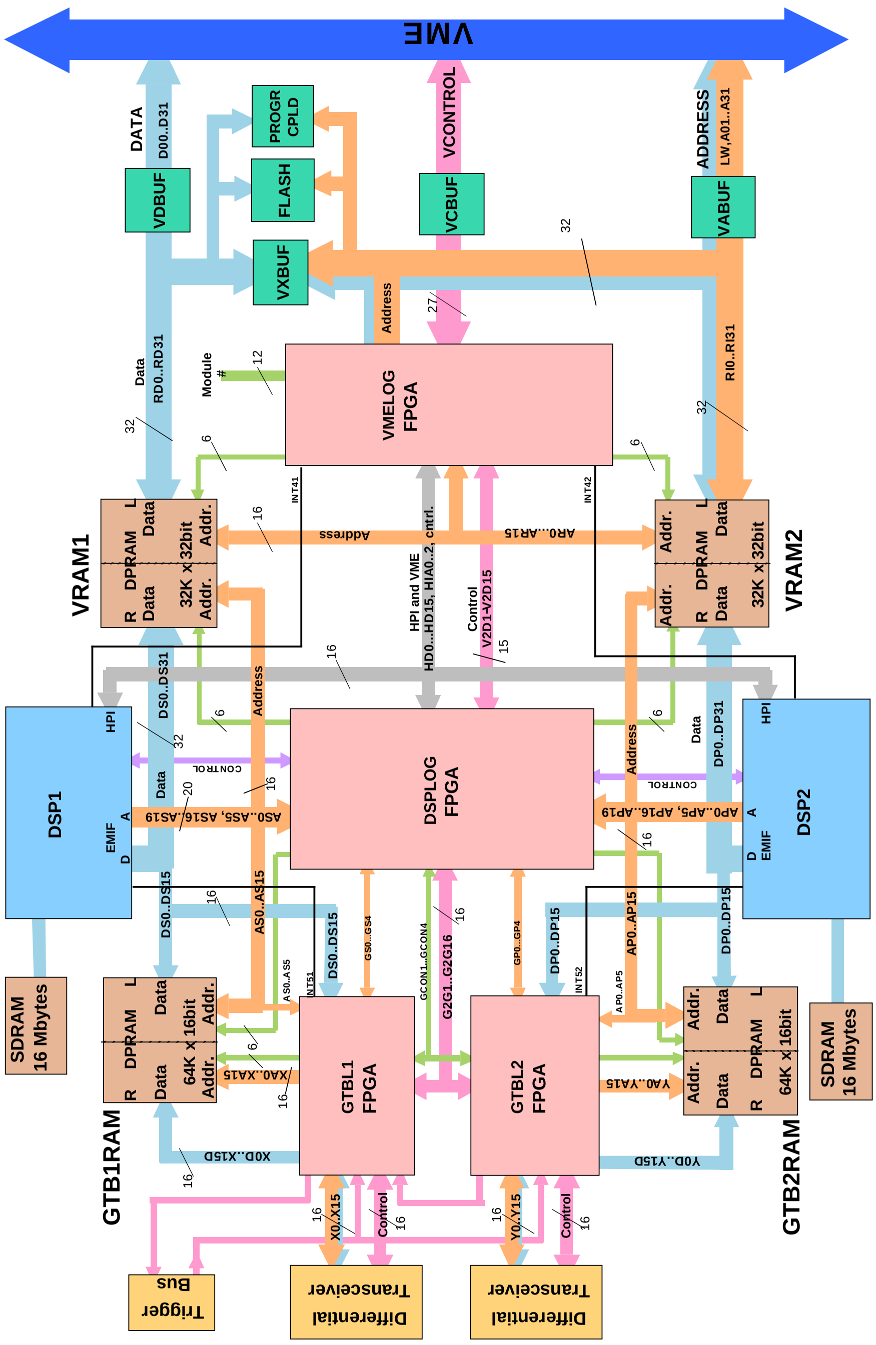


Figure 3.2: Schematic drawing of the SAM3 module. The important components like DSPs, RAMs, FPGAs and the address, data and control paths between them are plotted.

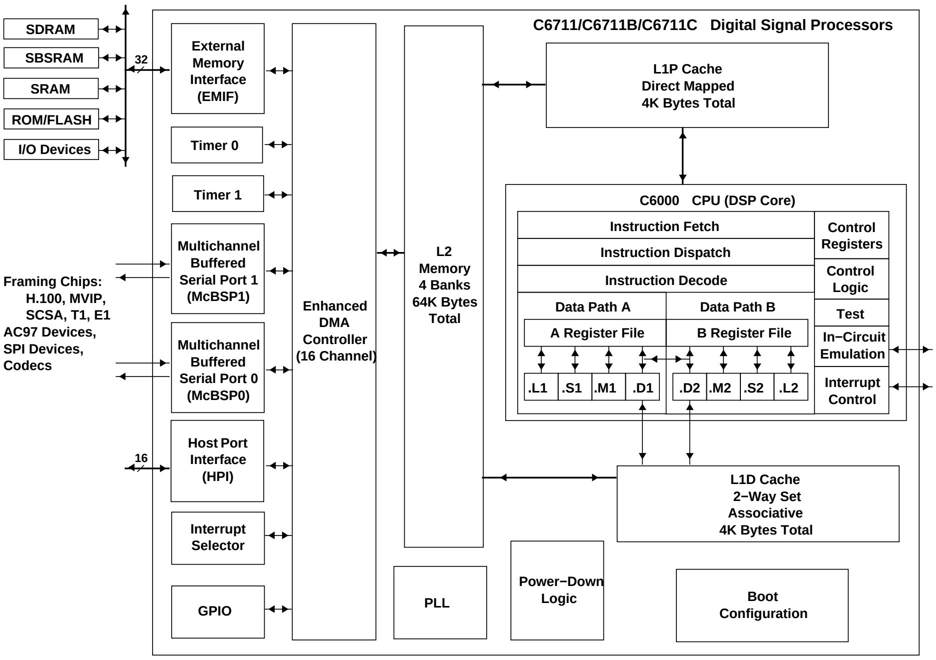

C6711/C6711B/C6711C Digital Signal Processors
SDRAM
SBSRAM
SRAM
ROM/FLASH
I/O Devices
32
External Memory Interface (EMIF)
Timer 0
Timer 1
Framing Chips: H.100, MVIP, SCSA, T1, E1 AC97 Devices, SPI Devices, Codecs
Multichannel Buffered Serial Port 1 (McBSP1)
Multichannel Buffered Serial Port 0 (McBSP0)
16
Host Port Interface (HPI)
Interrupt Selector
GPIO
Enhanced DMA Controller (16 Channel)
L2 Memory 4 Banks 64K Bytes Total
PLL
Power−Down Logic
L1P Cache Direct Mapped 4K Bytes Total
C6000 CPU (DSP Core)
Instruction Fetch
Instruction Dispatch
Instruction Decode
Data Path A
Data Path B
A Register File
B Register File
.L1
.S1
.M1
.D1
.D2
.M2
.S2
.L2
Control Registers
Control Logic
Test
In−Circuit Emulation
Interrupt Control
L1D Cache 2−Way Set Associative 4K Bytes Total
Boot Configuration

Figure 3.3: Schematic drawing of the C6711 DSP. The CPU core, the memories and the peripheral devices are plotted.

B). Three units are ALUs[11] (**.L**, **.S** and **.D**) while the fourth unit (**.M**) is a multiplier. The **.L** and **.S** units perform 32-bit arithmetic and logical operations as well as compare (**.L**) and shift (**.S**) operations. The .D unit is primarily used for address calculations and load/store operations. Fixed-point 16×16 bit multiply operations are done by the **.M** unit.

The general purpose registers are split into two groups of 16 registers (A0 – A15, B0 – B15), each register file being assigned to one of the data paths. They are used to keep data or data-address pointers. Some of the registers can also act as condition registers or for addressing purposes. Each functional unit can read/write directly from/to the register file connected to its own data path. In addition two cross paths exist which enable the functional units from one data path to access the register file belonging to the other data path.

The three instruction units are needed to execute a program on the DSP. The instruction fetch unit gets a 256 bit long instruction *fetch packet* from the program cache at the beginning of a processing cycle. This packet can contain one or more (up to 8) *execute packets* containing the 32-bit long in-

---

[11] Arithmetic Logic Units

structions which are to be simultaneously executed on the functional units. The instructions given in the execute packet are sent to the respective functional units by the instruction dispatch unit. Afterwards, the instruction decode unit extracts the operation to be performed and the operand's memory addresses.

Directly connected to the DSP core are the two 4 kB level-one cache memories, one for program code and one for data. In addition they are connected to a 64 kB large memory (shared between program and data) which can also be configured as level-two cache if desired.

Apart from the DSP core, the cache and internal memories the chip also contains some peripheral components for various purposes. Important ones are the enhanced DMA[12] controller (EDMA), the external memory interface (EMIF) and the host-port interface (HPI). Other components like timers, multichannel buffered serial ports, interrupt selector and power-down logic are not explained here.

- The EDMA controller allows to transfer data between different memories (internal and external) in the background without any help of the CPU. It has sixteen channels which can be independently programmed thus being able to supervise 16 different transfers. When a transfer is completed the controller can generate an interrupt which is sent to the CPU. This feature is used to copy data from GTBRAM into the DSP's internal RAM or from the DSP into the VRAM.

- External memories can be integrated into memory address space of the CPU by the EMIF. It supports different types of synchronous and asynchronous memories like SRAM[13], SBSRAM[14], SDRAM and ROM[15], flash memories and other devices. Each memory can be assigned to one of the four external memory spaces CE0 – CE3. Access parameters like read and write cycle times and the memory type can be set independently for the different memory spaces. On the SAM module the GTBRAM, SDRAM, VRAM and some control registers for communication with SSM and RIO3 are mapped into these memory spaces.

- The HPI provides a 16-bit parallel interface which enables a host processor to access the memory space of the DSP. The interface is configured through the HPI control register. After this has happened data can be read from or written to the memory space. The first step is

---

[12] Direct Memory Access

[13] Static RAM

[14] Synchronous Burst SRAM

[15] Read-Only Memory

to write an address to the HPI address register. Afterwards, the data word belonging to this address can be accessed via the HPI data register. In our case the HPI is used to directly control the SSM by the RIO3 (FADC calibration, testing) and to load the data reduction code into the internal memory of the DSP.

In order to have a fast data reduction and a fast data transport through the SAM module it will be necessary to utilize all the above mentioned features of the DSP in the best possible way. For the implementation of the data reduction algorithm on the DSP this means in particular that in order to achieve a high degree of parallelization many of the functional units should be used at the same time.

3.4 The Data Reduction Algorithm

The task of the data reduction algorithm is to extract the relevant information from the incoming raw data: Start time and integral of each peak in the FADC output are needed to determine position and energy loss of the corresponding hit in the CDC.

3.4.1 Working Principle of the Data Reduction

Figure 3.4 sketches the basic method of data reduction. Per drift chamber wire two signals (which can contain zero or more pulses) have to be processed (see panel (1)). Depending on the side of the CDC on which the preamplifier is located, the signals are phrased "upstream" (blue) or "downstream" (red).

The FADC digitizes these signals in time intervals of 10 ns and, due to the nonlinear characteristics of its internal preamplifier, decreases the pulse height (panel (2)) according to equation 3.1. When the SSM collects the FADC output data it performs a *zero suppression* which rejects samples below a certain threshold. In addition the data is scanned to find sections with more than three successive samples above this threshold. Whenever such a section is found it is accepted as a *hitlet* which can contain one or more peaks. Five *pre-samples* in front of the actual hitlet (panel (3)) are also stored for determination and subtraction of the FADC baseline by the DSP reduction program. When the SSM scanning procedure is finished, the resulting data are written into the GTBRAM of the SAM module.

Before starting the reduction procedure, the DSP retrieves the data from the GTBRAM into its internal RAM. The first step – after checking data

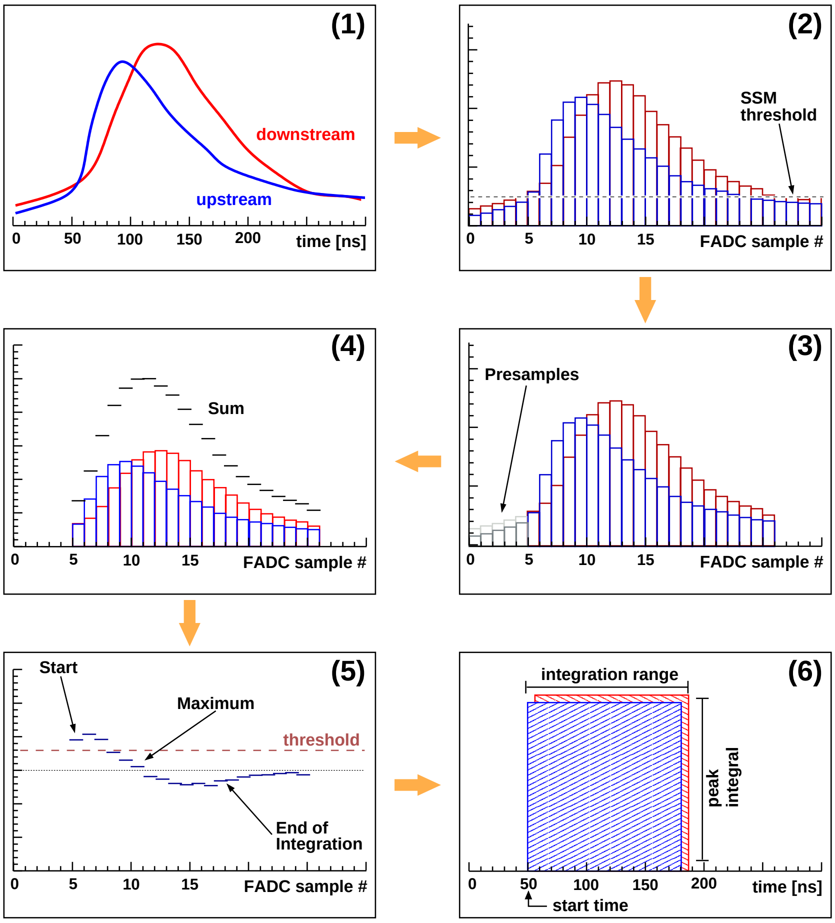


Figure 3.4: Illustration of the data reduction procedure. The output of the preamplifiers (1) is digitized by the FADCs (2). After application of the zero suppression and determination of the pre-samples (3) by the SSM, the data is linearized (4) by the DSP. Peaks are searched by looking at the derivative of the combined signal (5). Finally, start time and integral of the peaks are calculated (6).

integrity and hitlet length – is to apply the linearization correction to all samples in the hitlet and to calculate the sum of the upstream and downstream signals (panel (4)). In order to search the data for valid signal peaks the bin-by-bin difference of that sum is calculated. This ”derivative” is then analyzed further: If at least two consecutive bins are higher than some threshold (>4), a starting point for the rising slope of a peak is found. When the sign of the derivative changes from positive to negative, the peak's maximum is reached. Thereafter the integration endpoint is defined by an offset of seven more channels after the maximum and the peak search continues in the remaining part of the hitlet (see panel (5)). If a new peak is found to be within the integration range of the previous peak the integration is stopped at the beginning of the new peak.

Once all peaks in a hitlet are found, start time and peak integral are calculated separately for the upstream and the downstream side. Afterwards the FADC baseline correction and a correction for the truncation error for short integration ranges are applied. The corrected integrals are plotted according to their start time (position), integration range length (width) and integral value (height) (panel (6)). Together with the wire number and status quantities like the number of FADC overflow[16] samples, lengths of time and charge determination ranges, the results are stored in the VRAM. After the reduction procedure is finished, all output stored in the VRAM is collected by the RIO3.

3.4.2 Implementation of the Reduction Algorithm

The aim of the implementation of the algorithm is to make the best possible use of the different functional units of the DSP. An efficient parallel processing of the data is only possible if one has to repeat certain sets of instructions very often. Loops which contain only a few instructions (especially also very few load or store commands) are particularly advantageous.

The strategy used by the old data reduction algorithm was to execute all processing steps (linearization, differentiation) bin-by-bin during the peak search. This leads to large loops which are not suited for parallelization. For the new version of the reduction algorithm the various processing steps were split up. Linearization, differentiation, peak search, start time determination and peak integration are realized in separate loops executed one after another. In addition the copying of the SSM data from the GTBRAM into the DSP's internal RAM is accelerated by using a DMA transfer. The flow of the program includes the steps given in table 3.1.

After all the steps mentioned above had been programmed, the compiled version of the full reduction program had a size of about 20 kB. Therefore at least 32 kB of internal DSP memory could be reserved for the data of an event. Each wire delivers 2 bytes of data per 10 ns (8 bits for upstream and downstream sample values) and one DSP has to process the data of 48 wires.

---

[16] FADC overflows can occur due to ionizations by heavy fragments or as a consequence of a too high gain in the CDC. Sample values can then be larger than the maximum-allowed value of 253 (0xFD). Those samples are set to a value of 254 (0xFE) which indicates that there was an overflow. The value 255 (0xFF) is used as a control character to separate the hitlets.

step	phase	action	comment
1	**setup**	retrieve program options	set by RIO3 in communication block in VRAM
2		configure memories on SAM	set memory parameters (type, access options) of GTBRAM, VRAM and SDRAM for EMIF
3		set up linearization correction	calculate lookup table for fast linearization correction during data reduction
4		partition VRAM	define memory blocks for reduced or raw data
5		activate SSM	set GTBRAM address pointer to first free block
6		start infinite loop:	
a	**reduction**	increase DSP event number	compare to SSM event number later
b		empty data block in VRAM ?	previous data already retrieved by RIO3 ?
c		wait for SSM	FADC scanning finished ? ⇒ stop SSM
d		get SSM event number	compare to DSP event number
e		check event size & integrity	verify event size by checking end-of-data tag
f		copy event data	do DMA transfer from GTBRAM to DSP internal RAM
g		start SSM again	prepare SSM for next event
h		perform data reduction	write results to VRAM afterwards
i		message to RIO3	data ready in VRAM (block 0 or 1)
j		loop end	go back to **a**

Table 3.1: Program flow of the data reduction code for the DSP. The *setup* phase is passed once after the program is started. The *reduction* phase runs until a non-recoverable error occurs.

Therefore it is possible to read out about 340 FADC samples per wire (after zero suppression by the SSM). This corresponds to a time range of 3.4 μs. Due to the zero suppression one can effectively read out a much larger time range.

If one wants to analyze non-zero-suppressed data approximately 500 – 600 FADC channels have to be read out. This is necessary because the drift velocity for the gas mixture of CDC and Helitron is around 4 – 5 cm/μs and the longest drift path has a length of about 23 cm. To read out non-zero-suppressed data a special version of the reduction program was developed. The executable code has a size of less than 16 kB so that 48 kB of internal memory remain free for the data. In addition to the normal data reduction mode (RED) this version also includes a mode (RAW) which just copies the SSM data from GTBRAM to VRAM. This mode can be used for online analysis and monitoring purposes.

The code of the fast data reduction DSP program used in the S261 beam time is explained in appendix D.

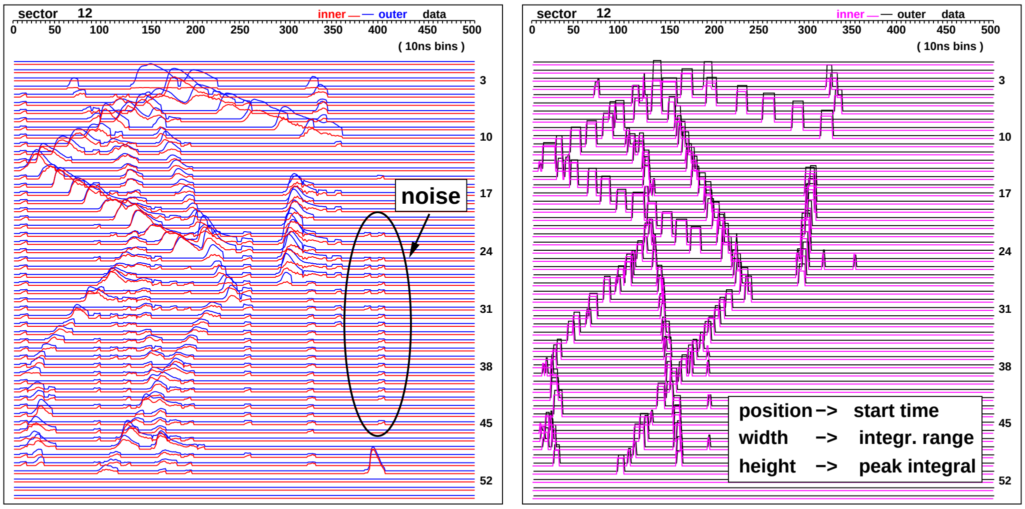


Figure 3.5: Example of the data reduction for a sector of the Helitron. The left panel shows the digitized pulses generated by several particles in an event on the 54 wires of the sector. The right panel shows the remaining pulses which were accepted by the data reduction (their width indicates the length of the integration range, their position the reconstructed starting time and their height the peak's integral).

3.5 Performance of the Data Acquisition

The reduction program used in the experiment featured the "token mode" data transfer between SSM and SAM, 32 kB of free internal RAM for SSM data and two separate memory blocks in in GTBRAM and VRAM which were used alternately to buffer the incoming and outgoing data. The free space in the internal RAM of the DSP allows to read out approximately 300 FADC samples per wire. In combination with the zero suppression done by the SSM this number is sufficiently large to allow the processing of all incoming events in Ni+Ni collisions.

A graphical example of the performance of the reduction code is given in figure 3.5. The left picture shows the raw data of an event for one sector of the Helitron. In this representation, the detected pulses are placed according to the wire number and to their arrival time. Several tracks are visible in the plot. Small peaks which occur on many neighbouring wires at the same time indicate correlated noise. The right picture presents the remaining data after application of the reduction algorithm. Position, width and height of the peaks now represent start time, integration range and peak integral, respectively (cf. figure 3.4). While the tracks have been detected, the correlated noise was rejected to a large extent.

The rate at which events could be processed by the new DAQ system including data reduction was around 120-150 Hz depending on the noise picked up by the drift chambers (although the program suppresses the noise signals in the output it has to identify them first). The old acquisition system was able to reduce the data at a rate of some 10–30 Hz. Raw data without any reduction are now recorded at a rate of 10-20 Hz. This rate is not limited by the DSP but by the bandwidth (100 Mbit) of the LAN interface of the RIO3.

The average event size each DSP had to process was about 15–20 kB which corresponds to 3–5 hits per wire with approximately 30 samples per hit for 48 wires per DSP (upstream and downstream signals) and noise-induced fake signals. After data reduction the event size had decreased to about 2–4 kB corresponding to a reduction factor of approximately 5–7.

With this performance of the new DAQ system high statistics experiments have become feasible. In the 2003 Ni+Ni experiment more than $110{\cdot}10^6$ collisions could be recorded. This number corresponds to more than 6 TB of raw data which also constitutes a challenge for the calibration and analysis efforts.

Chapter 4

Data Analysis

The first step of the data analysis is to characterize events in terms of centrality, background contributions and reaction geometry and to identify detected particles.

The centrality selection has to be made in order to enrich certain types of events in the data and to be able to compare the results to those of previous experiments. Imposing constraints to the vertex position and the start counter information reduces the background caused by interactions which did not occur in the target or were caused by multiple events. Particles are identified through their mass which is determined from mean energy loss and momentum of the corresponding track. Due to its geometry and to certain kinematical constraints, the detector (CDC) has a limited acceptance which has to be taken into account when interpreting the results. Finally, for the investigation of collective effects, the reaction plane needs to be determined for each event. These steps will be described in the following sections.

4.1 Centrality Selection

The centrality of a collision is linked to the multiplicity of emitted particles [91]. As collisions become more central (the impact parameter decreases), more particles participate in the reaction and gain transverse momentum which makes them more likely to be detected.

The multiplicity of charged particles measured in the forward Plastic Wall (PMUL) was used as an online trigger for central events. Two trigger types are defined, *central* (PMUL > 22) and *medium-bias* (PMUL > 16). The central trigger comprised approximately 20 % of all events. The total geometrical cross section for Ni+Ni collisions can be estimated using a sharp

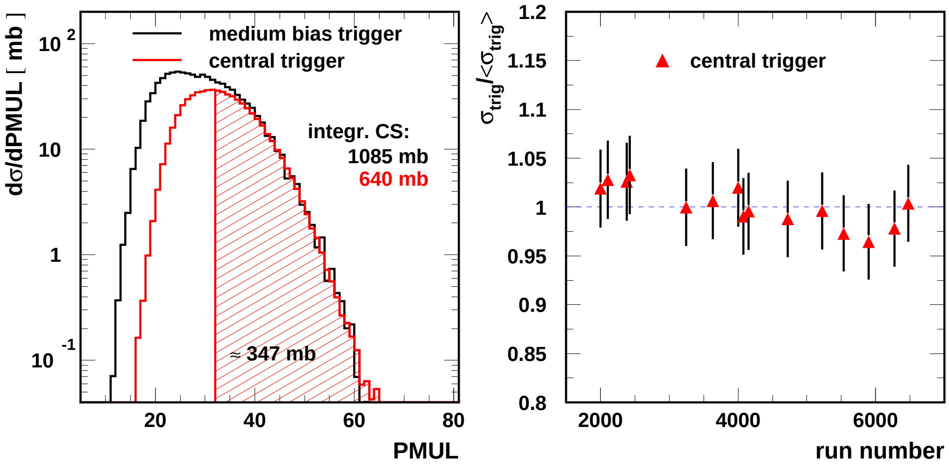


Figure 4.1: Triggered reaction cross sections as function of the Plastic Wall multiplicity PMUL for medium bias and central trigger (left) and relative fluctuations of the triggered cross sections as function of the run number (right).

cutoff model:

$$\sigma = \pi \cdot b_{max}^2 = \pi \cdot (2 \cdot r_{Ni})^2 = \pi \cdot \left(2 \cdot r_0 \cdot \sqrt[3]{A_{Ni}}\right)^2 \tag{4.1}$$

With $r_0 = 1.2$ fm one finds a value of 2711 mb for the total geometrical cross section.

The scaler information (N_{beam}, $N_{\text{trig.}}$, $ds_{\text{tot.}}$), the hardware down-scale factor ($ds_{\text{trig.}}$) and the target areal density ($n_{\text{targ.}}$) are needed for the determination of the measured reaction cross section. With the formula given in section 2.3 one calculates the cross section σ_{exp} for a certain trigger class. The left picture of figure 4.1 shows the Plastic Wall multiplicity distributions for medium-bias and central trigger. Integration of the histograms yields a cross section of about 1100 mb for the medium-bias trigger and 640 mb for the central trigger. The relative fluctuation of the triggered cross section for the central trigger is displayed as a function of the run number in the right picture of figure 4.1. The central cross section is found to vary within a range of less than ±5 %.

4.2 Rejection of Background Events

Background events recorded in the experiment have two major sources. The first one are events which do not originate from the target but from inter-

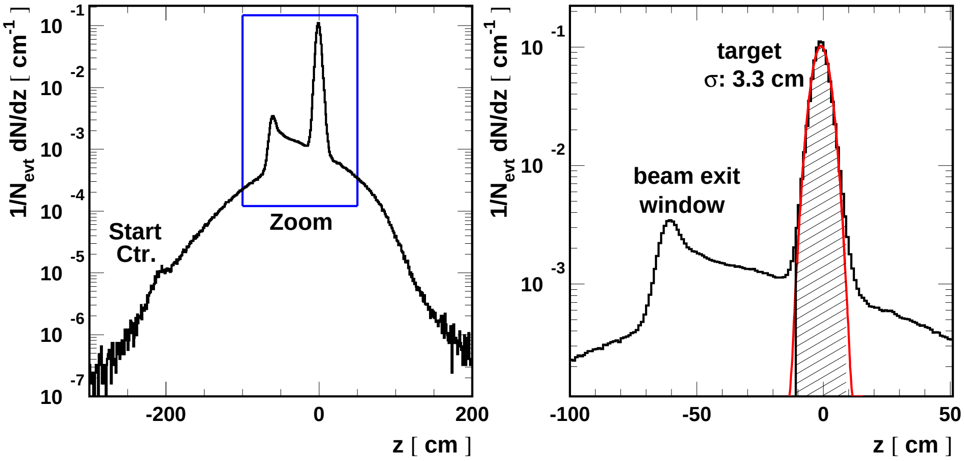


Figure 4.2: Distribution of the z component of the event vertex (left) and zoomed view of the target region (right).

actions in the exit window of the beam pipe or from collisions between ions and gas molecules in the target area. The second source are so-called pileup events where two or more ions reach the target at about the same time.

The vertex position of an event is determined by extrapolating back the measured tracks towards the target in order to find a point which fits best as an event vertex. The different position resolution of hits in the x–y and r–z planes is also reflected in the accuracy of the vertex position. While the transverse resolution is about 1.4 mm, the longitudinal resolution is of the order of 3.3 cm. Due to the fact that the ion beam is focused in the x–y plane, background events are suppressed by constraining the z coordinate of the event vertex.

The left picture of figure 4.2 shows the normalized distribution of the z coordinate of the event vertex ranging from 3 m in front of to 2 m behind the target. It exhibits a pronounced peak at the target position sitting on an underlying background of events which are not coming from the target but were accepted by the trigger. In addition to the target peak, the start counter (at about -2 m) and the exit window of the beam pipe (at -60 cm) are visible.

The right picture shows an enlarged view of the target region. The peak at -63 cm indicates background events arising from interactions in the exit window of the beam pipe and the continuum is the result of collisions between Ni ions and He atoms in the helium-filled gas bag. In order to suppress this

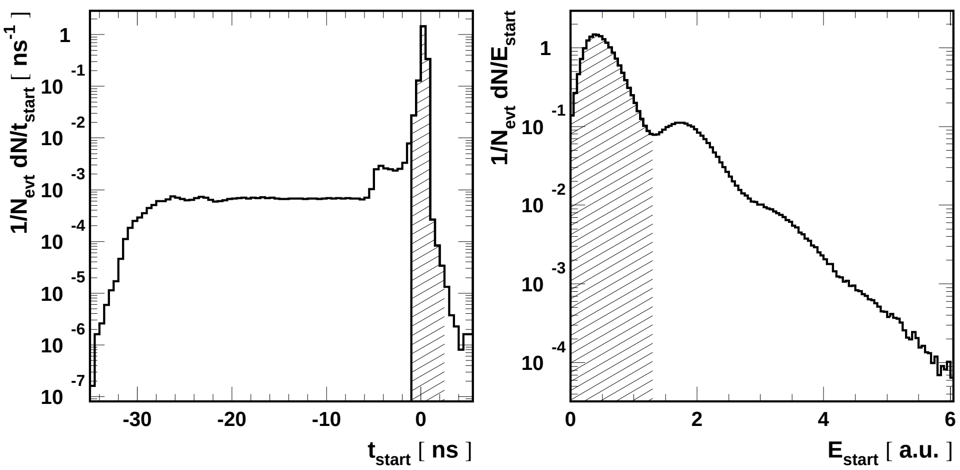


Figure 4.3: Start counter time (left) and energy spectra (right).

background to a large extent a cut is applied which only accepts events coming from within a $\pm 3\cdot\sigma$ interval (= ± 10 cm, hatched area) around the the nominal z position of the target. Thus approximately 86 % of all events are accepted. In the x-y plane all vertices within the interval -1.5 cm to +1.5 cm (in both directions) are accepted. The fraction of background in the remaining events is less than 2 %.

The start counter information is not only used to count the incoming ions for the calculation of the trigger cross section but also to discard background events by constraining the start time and energy loss signal. With $4\text{-}5\cdot 10^6$ ions per spill and a spill length of 10 s the mean time difference between two successive ions is about 2-2.5 μs. Although the detection efficiency of the start counter is much better than 99 % some ions remain undetected. Therefore it can happen that one of the two ions is not detected but reacts in the target while the other one is detected but does not produce a reaction. If both ions arrive within the time interval of the start counter gate (100 ns), the start time of the corresponding event is wrong. Thus, no correct time-of-flight and drift time information is available for the particles in that event. The effects of this occurrence are visible in the start counter time spectrum plotted in the left panel of figure 4.3. After calibration the start time of valid events should be zero in this spectrum. A preselection cut of $-1 < t_{start} < +2$ indicated by the hatched area in the plot discards roughly 3 % of all events.

When two (or more) ions pass the start counter at about the same time, a so-called 'double hit' can occur in the target. This is problematic for thick targets or analyses of rare particles because it could cause more background or even worse, a fake signal. A double or multiple hit can be seen in the start counter energy loss spectrum presented in figure 4.3. The first peak represents the case when a single ion passed the counter. The second peak and the following continuum mark events when more than one ion passed by at the same time. When requesting an energy below 1.4 [a.u.] about 14 % of all events are rejected. Due to the fact that a 1.5% target was used in the experiment the probability for a double hit is only 0.02 %. Therefore no cut is applied to the start counter energy information for the reconstruction of relatively common strange particles like Λ^0 or K^0_S. For rare strange particles like Ξ^- or $\Sigma^-(1385)$ double hits have to be discarded.

preparation step	**no. of events**
available from DSTs	87 254 206
after vertex cuts (x, y, z)	75 303 307
after start counter cuts (t_S, E_S)	72 674 104

Table 4.1: Number of events at different stages of the analysis.

The effect of the quality cuts can be seen from table 4.1. Confining the z position of the event's vertex to an interval of ±10 cm around the target position decreases the number of available events from 87 millions to 75 millions. The start counter cuts reject another 3 million events.

4.3 Particle Identification

For the reconstruction of both, neutral strange particles (K^0_S, Λ^0) and cascades (Ξ^-, $\Sigma^-(1385)$), their decay products have to be identified.

The CDC provides information about the track's curvature r_c, its polar angle ϑ_{lab}, the mean energy loss $\langle dE/dx \rangle$ and the closest approach to the primary vertex, d_0. The sign of the particle charge and its transverse momentum per unit charge are given by the curvature. Using r_c (in [cm]), the magnetic field B (= 0.6 T) and the vacuum light velocity c one finds

$$\frac{p_t}{|q|} = |r_c| \cdot B \cdot c \cdot 10^{-9} \quad \text{in [GeV/c]}. \tag{4.2}$$

The total momentum can then be calculated from the transverse momentum and the polar angle.

The mean energy loss of a track is determined from the contributions by all hits it consists of. The measured charge deposited by a hit represents the

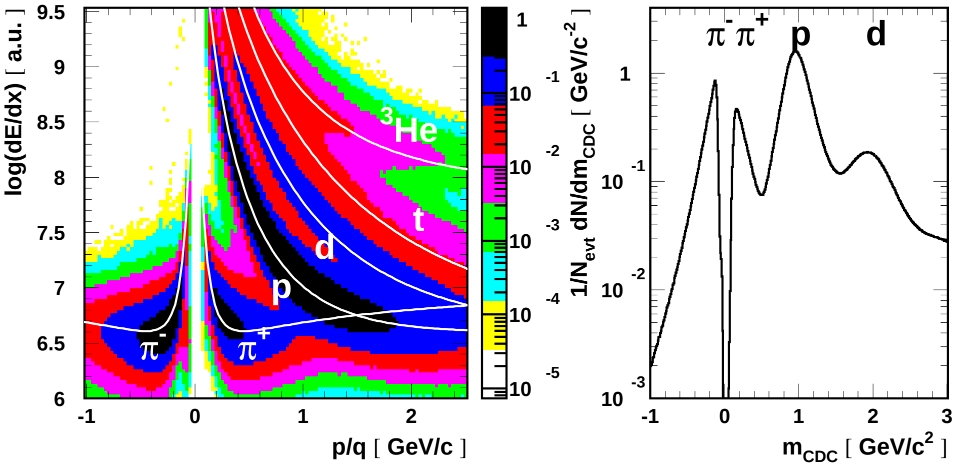


Figure 4.4: Logarithm of the energy loss of particles in the CDC as function of their momentum/charge (left) and the mass spectrum extracted with the Bethe-Bloch formula (right).

amount of energy the particle lost while passing a drift cell. The spectrum of these energy-loss values resembles a Landau distribution [81] which has a long tail towards high values of dE/dx. This can cause systematically higher values of the mean energy loss due to statistical fluctuations. A *truncated mean* method is applied using a lower threshold for the energy loss and, in addition, discarding the 30% largest values. This results in a more reliable measurement of the mean energy loss.

The event vertex position in the transverse plane is obtained in an iterative procedure by first finding a vertex which best fits all tracks and then discarding the most distant tracks until the position converges. Afterwards the distant of closest approach, d_0, is calculated for each track with respect to the primary vertex.

The particle mass is determined from the mean energy loss $\langle dE/dx \rangle$ and the momentum per unit charge p/q which are linked by the Bethe-Bloch formula (see appendix B). A typical two-dimensional distribution of the logarithm of the mean energy loss and the momentum per charge is plotted in the left panel of figure 4.4. Parameterizations of the Bethe-Bloch formula for different particle species are indicated by the white lines in the picture. Pions, protons and deuterons can easily be distinguished. Heavier particles like tritons and ^{3}He are also visible. The resulting CDC mass spectrum is shown in the right picture of figure 4.4. A certain particle species is selected

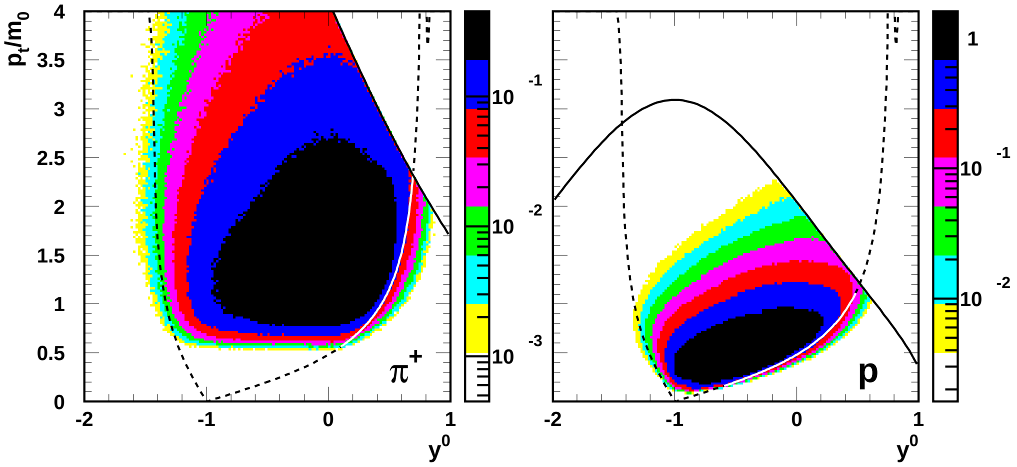


Figure 4.5: Acceptance for π^+ (left) and protons (right) detected in the CDC. The dashed lines indicate the geometrical acceptance defined by the polar angle range in the laboratory system. The solid lines demonstrate the limitation by a cut on the laboratory momentum.

by applying a mass range cut in the analysis.

4.4 Detector Acceptance

The CDC covers the full azimuthal range but the acceptance for the polar angle is limited from 23° to 113°. Altogether, this corresponds to a geometrical acceptance of about 70%.

Figure 4.5 gives two examples of the acceptance of the CDC for different particle species. The left panel shows the distribution of π^+ particles according to their normalized transverse momentum p_t/m_0 and the scaled rapidity y^0 (see appendix C). The geometrical acceptance is illustrated by the dashed lines. The distribution exhibits a good coverage from target rapidity towards mid-rapidity and beyond. In order to prevent a contamination by protons when determining the masses from the energy loss, an upper limit for the momentum in the laboratory frame was set at 820 MeV/c, indicated by the solid line. Due to the lower momentum threshold of 80 MeV/c which is needed to reject particles spiraling in the CDC no entries are found below $p_t/m_0 \approx 0.6$. The corresponding spectrum for the π^- is similar to that one of the π^+ with the exception that no upper momentum cut is needed.

The proton distribution is shown in the right panel. Due to the much higher rest mass its spread is small compared to that of the pions and the geometry of the CDC has a stronger effect on the acceptance around mid-rapidity. The upper momentum limit marked by the solid line in this case was about 2.9 GeV/c to prevent a contamination by deuterons.

4.5 Reaction Plane Reconstruction

The study of collective effects in heavy-ion collisions is a tool to measure the properties of nuclear matter. In order to investigate these effects the emission of particles has to be studied with respect to the reaction plane of each event. The reaction plane is defined by the impact parameter ($\vec{b}$) and the beam direction (z).

The orientation of the impact parameter cannot be measured directly in the experiment but has to be determined from experimental observables. The method used here was proposed by Danielewicz and Odyniec [92]. The angle ϕ_R between the direction of the impact parameter and the x axis of the detector's coordinate system is calculated by summing the transverse momenta of all particles of an event:

$$\vec{Q} = \begin{pmatrix} Q\cos\phi_R \\ Q\sin\phi_R \end{pmatrix} = \sum_{i=1}^{N} w_i \cdot \vec{p}_{t,i} \tag{4.3}$$

The weighting factor w_i is chosen in the following way:

$$w_i = \begin{cases} +1 & \text{for } y^0 > \Delta y \\ 0 & \text{for } -\Delta y < y^0 < \Delta y \\ -1 & \text{for } y^0 < -\Delta y \end{cases} \tag{4.4}$$

where y^0 denotes the scaled rapidity explained in appendix C. Particles from the interval $[-\Delta y, \Delta y]$ around mid-rapidity do not provide information about the reaction geometry and are excluded from the sum. A value of $\Delta y = 0.2$ was used in the analysis.

From all particles detected in backward hemisphere by the CDC+Barrel combination, those which were produced (mainly $\pi^\pm$ but also $K^\pm$) are not taken into account in the sum since they do not carry any information about the initial reaction geometry. Particles detected in the forward hemisphere by the Plastic Wall cannot be identified like in the CDC. In this case, all detected particles are included in the sum.

A sample distribution of reaction plane angles for events containing K^0_S candidates is shown in figure 4.6, left picture. For positive values of ϕ_R a slight excess can be observed which is peaked at around 90°. This could be

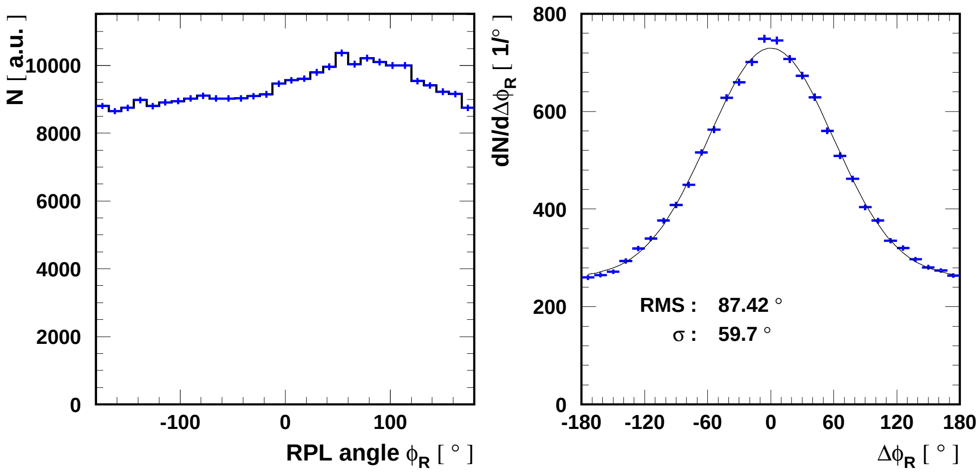


Figure 4.6: Distribution of reaction plane angles for K_S^0 candidates (left) and of the resolution $\Delta\phi_R = \phi_1 - \phi_2$ (right).

the result of fake tracks with high transverse momentum caused by noise in one or more CDC sectors at the corresponding position.

The quality of the reaction plane determination can be estimated by a method also introduced in [92]. All particles of the event are randomly assigned to one of two sub-events. The reaction plane angles $\phi_{1,2}$ are then calculated with the procedure explained before. Their difference $\Delta\phi_R = \phi_1 - \phi_2$ is a measure of the reaction plane resolution which is plotted in the right picture of figure 4.6. The RMS value of 87.4° is comparable to the one given in [93] for the same system at the same energy (87.30°). If one fits a Gaussian plus a constant offset to the distribution a width of 60° is found. Thus even in central events a reaction plane can still be found with a fair accuracy.

Due to fluctuations in the particle multiplicity and the transverse momentum resolution the reconstructed reaction plane will differ from the true reaction plane by an angle $\Delta\phi$. To correct the Fourier coefficients $v_n = \langle \cos n\varphi \rangle$ (see chapter 1) for the flow analysis the so-called Ollitrault correction [94] is applied: If φ is the angle between a particle and the reconstructed reaction plane and ϕ the angle between that particle and the true reaction plane one finds $\varphi = \phi - \Delta\phi$. With the assumption that $\Delta\phi$ and ϕ are independent and when averaging over many events, the measured and the true Fourier coefficients are related:

$$\langle \cos n\varphi \rangle = \langle \cos n\phi \rangle \langle \cos n\Delta\phi \rangle \, . \tag{4.5}$$

The correction factor $\langle\cos n\Delta\phi\rangle$ for the n-th Fourier coefficient can be calculated from a parameter χ. This parameter is a function of $\langle\cos\Delta\phi_R\rangle$ and can be determined with the equation

$$\langle\cos\Delta\phi_r\rangle = \frac{\pi}{8}\chi^2 e^{-\chi^2/2}\left[I_0\left(\frac{\chi^2}{4}\right) + I_1\left(\frac{\chi^2}{4}\right)\right]^2 , \tag{4.6}$$

using the modified Bessel functions I_0 and I_1. Definitions and tables of these functions can be found e.g. in [95]. For the n-th correction factor one finds

$$\langle\cos n\Delta\phi\rangle = \frac{\sqrt{\pi}}{2}\chi e^{-\chi^2/2}\left[I_{\frac{n-1}{2}}\left(\frac{\chi^2}{2}\right) + I_{\frac{n+1}{2}}\left(\frac{\chi^2}{2}\right)\right] . \tag{4.7}$$

flow observable	**correction factor**	
v_1	$\langle\cos\Delta\phi\rangle^{-1}$	1.54
v_2	$\langle\cos 2\Delta\phi\rangle^{-1}$	3.32

Table 4.2: Correction factors for v_1 and v_2 determined with Ollitrault's method presented in [94, 96].

From the distribution of $\cos\Delta\phi_R$ (not shown) a mean value of 0.250 can be extracted. Using equation 4.6 the parameter χ is determined. The correction factors for v_1 and v_2 calculated with equation 4.7 are given in table 4.2.

Chapter 5

Particle Reconstruction

While charged kaons can be identified directly by combining CDC and Barrel information, particles like K^0, Λ^0, $\Sigma^-(1385)$ and Ξ^- have to be reconstructed from their decay products. Weakly decaying strange particles have lifetimes of the order of 10^{-10} s. This means that a certain fraction of them decays at some distance to the primary vertex. By combining all possible decay particles in an event, secondary vertices are searched and the strange particles are identified by the invariant mass method. Combinations of uncorrelated particles produce a background which has to be subtracted before extracting physical quantities from the reconstructed strange particle candidates. This background can be determined by combining particles from different events.

At first, decay properties of the strange particles are given and the criteria which are used to select their decay products are explained. The methods to reconstruct both, neutral strange particles (K^0,Λ^0) and cascades (Ξ^-,$\Sigma^-(1385)$) are described and a method to increase their momentum resolution are presented. The determination of the combinatorial background by the event-mixing technique and the procedure to subtract this background from measured spectra are introduced. Afterwards, the background corrected invariant mass spectra and phase space distributions are presented for the reconstructed K^0 and Λ^0 candidates. The status of the search for Ξ^- and $\Sigma^-(1385)$ will be given in chapter 9.

5.1 Decay Properties of Strange Particles

For strange particle detection with FOPI, only decay channels which solely comprise charged particles are of interest. In addition, certain decay properties are needed in order to collect sufficient statistics:

1. The branching ratio (BR) of the decay channels should be of the order of 10 % or more. Otherwise even 10^8 events will not contain a large number of strange particles.

2. Weakly decaying particles which have lifetimes of approximately 10^{-10} s can survive long enough to produce a secondary vertex. By rejecting all particles coming from the primary vertex, the background can be reduced significantly.

3. In this work, the search for strange particle candidates is done via their invariant mass. Weakly decaying particles will produce a pronounced peak in this spectrum and are therefore identified easily. Due to the timescale of the strong decay ($\approx 10^{-23}$ s) particles like the $\Sigma^-(1385)$ have large widths of several 10 MeV. Hence, they are hard to identify.

Properties of the particles under consideration are given in table 5.1. The product of light velocity and mean lifetime ($c \cdot \tau$) gives an estimate of the distance between primary and secondary vertex.

particle	**m [GeV/c^2]**	**$c \cdot \tau$ [cm]**	**decay mode**	**BR**
K_S^0	0.497	2.68	$K_S^0 \to \pi^+ \pi^-$	68.6 %
Λ^0	1.116	7.89	$\Lambda^0 \to p\, \pi^-$	63.9 %
Ξ^-	1.321	4.91	$\Xi^- \to \Lambda^0 \pi^-$	99.9 %
$\Sigma^-(1385)$	1.387	$\Gamma = 39.4$ MeV	$\Sigma^-(1385) \to \Lambda^0 \pi^-$	88 %

Table 5.1: Properties of strange particles suitable for the detection in the CDC. The values are taken from [97].

The quark contents and schematic illustrations of the relevant decay channels are provided in figure 5.1.

- Due to K^0-$\overline{K^0}$ mixing, 50 % of the produced K^0 mesons are K_S^0 and 50 % are K_L^0. The K_L^0 cannot be observed in the detector because of its large lifetime ($c \cdot \tau = 15.51$ m) and the fact that its final state includes uncharged particles (π^0). The K_S^0 on the other hand decays into π^- and π^+ with a BR of 68.6 %. Its mean lifetime is large enough for a sizable fraction of the K_S^0 to produce a secondary vertex.

- The Λ^0 baryon decays into π^- and p with a BR of 63.9 % and a mean lifetime which is about a factor 3 larger compared to that of the K_S^0. It can also be identified by searching for a secondary vertex.

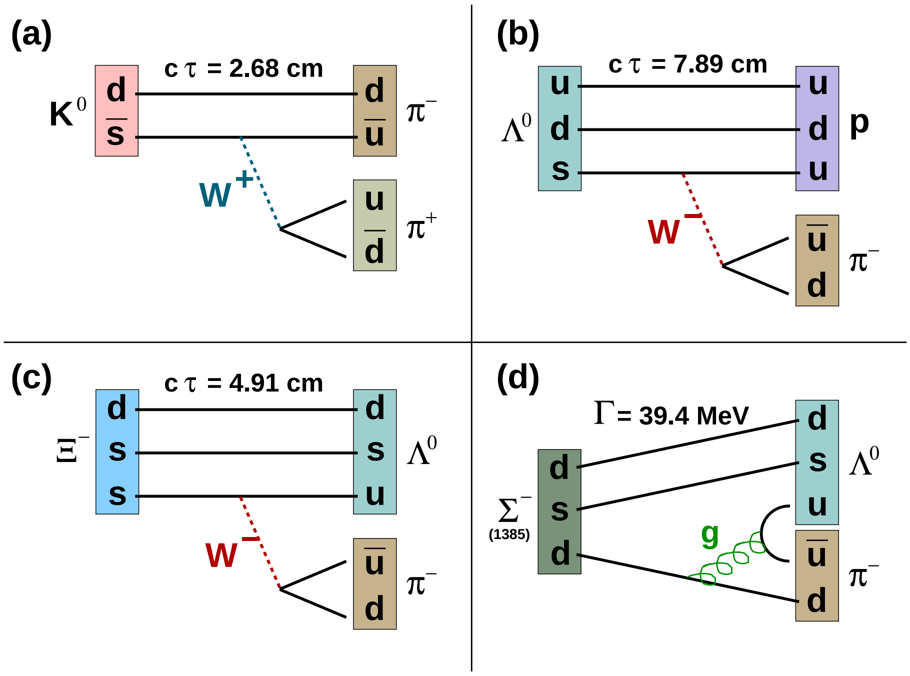


Figure 5.1: Schematic illustrations of strange particle decay channels relevant for this work. K^0 (a), Λ^0 (b) and Ξ^- (c) decay weakly at a timescale of 10^{-10} s. In the strong decay of the $\Sigma^-(1385)$ (d) with a timescale of 10^{-23} s strangeness is conserved. While the K^0 and Λ^0 decays are single processes with two charged particles in the final state, the Ξ^- and $\Sigma^-(1385)$ decays are two-step processes with three charged decay particles.

- The double strange Ξ^- baryon is a *cascade* particle which decays into π^- and Λ^0. Subsequently the Λ^0 decays in the same way as mentioned above. Therefore, the reconstruction proceeds in two steps, first searching for the secondary vertex of the Λ^0 decay and afterwards for that of the Ξ^- decay.

- Although the $\Sigma^-(1385)$ is a single-strange baryon, its decay products correspond to those of the Ξ^-. The first step now is a strong decay into π^- and Λ^0 where strangeness is conserved. Thus the lifetime for this step ($\Gamma = 39.4$ MeV) is of the order of 10^{-23} s. This means that the first decay vertex coincides with the primary vertex and the decay products cannot be distinguished from primarily produced particles. The overall BR is about 56 %.

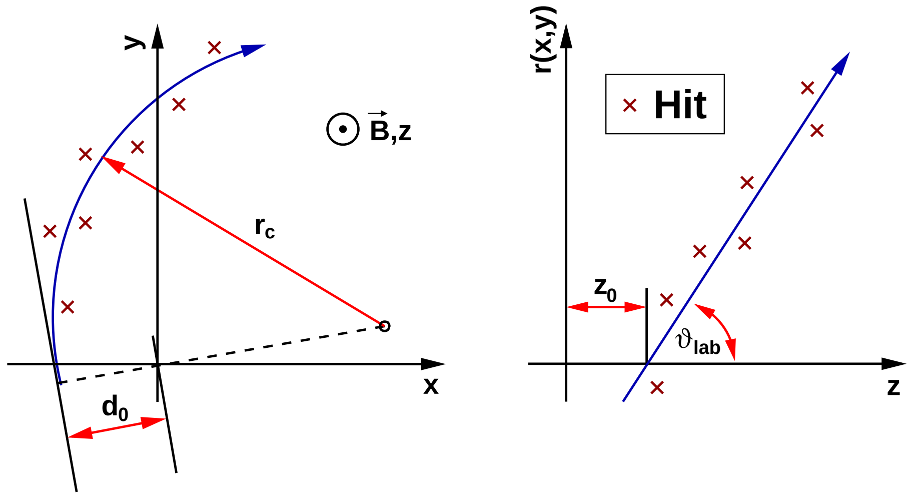


Figure 5.2: Determination of the shortest distance d_0 to the primary vertex in the transverse plane (left panel) and in the z (beam) direction (z_0, right panel).

5.2 Decay Particle Selection

Selecting the decay products is the first step in the strange particle reconstruction procedure. From the information the CDC provides for each track four quantities are used here: The track's curvature (r_c), the number of hits on a track, the CDC mass (m_{CDC}) and the shortest distance to the primary vertex in the transverse plane (d_0). The quantities are illustrated in figure 5.2 and the cut values for (pre-)selecting the particles are given in table 5.2. The final analysis cuts for all particles can be found in appendix E.

- **Curvature (r_c):** The curvature of a track is a measure of the transverse momentum of a particle (see eq. 4.2). A radius of at least 44.4 cm corresponding to a transverse momentum of about 80 MeV/c is requested to reject particles spiraling in the CDC due to their low momenta. The determination of the correct charge would be impossible in this case. A transverse momentum of 100 MeV/c ($r_c \approx 55.5$ cm) is needed for a track to produce a hit in the TOF-Barrel detector.

- **Hit multiplicity:** The hit multiplicity indicates how many hits belong to a certain track and is therefore approximately proportional to its length. A minimum hit multiplicity cut is applied to discard short tracks which probably are parts of other tracks the track-finding algorithm could not join.

- **CDC mass (m_{CDC}):** A selection of the particle species (π, p) is achieved by applying a mass window to the CDC mass spectrum (cf. figure 4.4, left plot). This quantity is derived from the mean energy loss and the momentum per charge of a track (see chapter 4). The adopted mass range for pions is between 0 and 600 MeV/c^2 and between 0.5 to 1.5 GeV/c^2 for protons.

- **Transverse and distance-to-vertex (d_0):** When the event vertex position has been determined, the distance of each track to this vertex is calculated in the transverse and longitudinal direction. A lower threshold for d_0 is used to select particles coming from a secondary vertex.

mother particle	**decay particles**		**r_c in [cm]**	**hit mult.**	**m_{CDC} in [GeV/c^2]**	**$\lvert d_0 \rvert$ in [cm]**
K_S^0	π^-		> 50.0	> 23	$0.04 < \cdots < 0.7$	$0.7 < \cdots < 20.$
	π^+		> 50.0	> 23	$0.04 < \cdots < 0.7$	$0.7 < \cdots < 20.$
Λ^0	π^-		> 44.4	> 24	$0.05 < \cdots < 0.7$	$1.2 < \cdots < 40.$
	p		> 55.5	> 24	$0.5 < \cdots < 1.5$	$0.55 < \cdots < 40.$
Ξ^-	Λ^0	π^-	> 50.0	> 24	$0.0 < \cdots < 0.55$	$0.5 < \cdots < 30.$
		p	> 50.0	> 24	$0.5 < \cdots < 1.5$	$0.4 < \cdots < 30.$
	π^-		> 44.4	> 24	$0.0 < \cdots < 0.55$	$0.3 < \cdots < 30.$
$\Sigma^-(1385)$	Λ^0	π^-	> 44.4	> 24	$0.05 < \cdots < 0.7$	$1.0 < \cdots < 40.$
		p	> 55.5	> 24	$0.5 < \cdots < 1.5$	$0.5 < \cdots < 40.$
	π^-		> 55.5	> 24	$0.05 < \cdots < 0.7$	$0.0 < \cdots < 1.2$

Table 5.2: Preselection cuts for final state particles used in the reconstruction. Detailed lists of the final cuts are provided in appendix E.

5.3 Reconstruction Method for K^0 and Λ^0

Neutral strange particles like K^0 and Λ^0 are identified by searching for intersection points of π^--π^+ or π^--proton track pairs in the transverse plane of the CDC. If an intersection point is found, momentum and invariant mass of a possible strange particle are calculated using the information from the decay products. The momentum resolution for the strange particle can be improved by refitting the decay particle tracks at the intersection point.

The geometrical reconstruction of secondary vertices is depicted in figure 5.3. First, the distance d_{21} between the two centers of the circles (M_1, M_2)

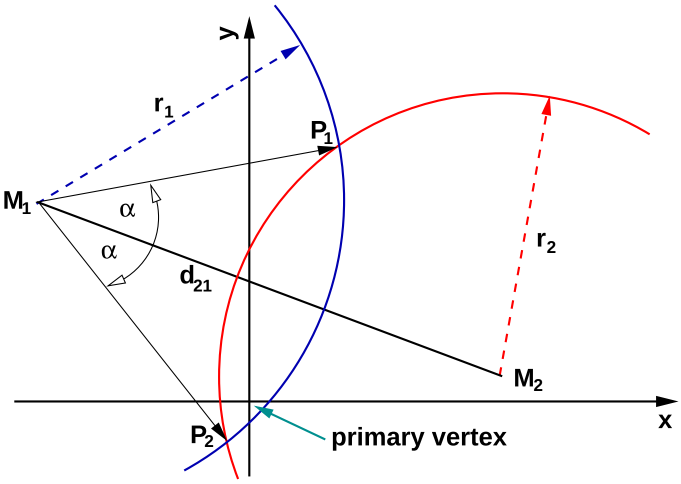


Figure 5.3: Determination of the secondary vertex position for K^0 and Λ^0. The blue and red lines indicate the π^- and π^+/proton tracks, respectively. The geometrical reconstruction is explained in the text.

fitted to the tracks is evaluated and compared to the sum of the radii (r_1, r_2). In the case of $d_{21} > r_1 + r_2$ no intersection points exist. Therefore, such a pair of tracks is discarded and the search is continued with another pair. If $d_{21} \leq r_1 + r_2$, one or two intersection points can be found. However, the case of one intersection point has no practical relevance. When α has been determined, the coordinates of the intersection points in the transverse plane are calculated and the geometrical reconstruction is complete.

In a next step, cut quantities and properties of the neutral strange particle candidate are determined for each intersection point (see fig. 5.4). The x and y coordinates of the secondary vertex are used to calculate the transverse distance r_t to the primary vertex. The azimuthal angle ϕ_t and the longitudinal positions (z_1, z_2) of the two tracks at the radial distance of the secondary vertex are also determined. In order to reduce the background coming from primary particles and to reject secondary vertices far away from the primary vertex ($>$ 5–6$\cdot$ $c\tau$), a lower and an upper threshold for r_t are applied. In addition, track pairs with a large distance in the longitudinal direction are suppressed by a cut on Δz.

The 4-momenta of the decay particles are calculated at the position of the secondary vertex using curvature, polar angle and the azimuthal angle at this point. Subsequently, the momenta (p_t, p_{tot}) of the strange particle candidate are determined and used to derive the azimuthal and polar angles (ϕ_p, ϑ_p). Further cut quantities like the *pointing angle* $\Delta\phi$ and the shortest distances to the primary vertex in the transverse and longitudinal direction are calculated.

Finally, the invariant mass m_{inv} of the candidate is determined according to

$$m_{inv} = \sqrt{(E_1 + E_2)^2 - (\vec{p}_1 + \vec{p}_2)^2} = \sqrt{E_{tot}^2 - \vec{p}_{tot}^2}\ , \qquad (5.1)$$

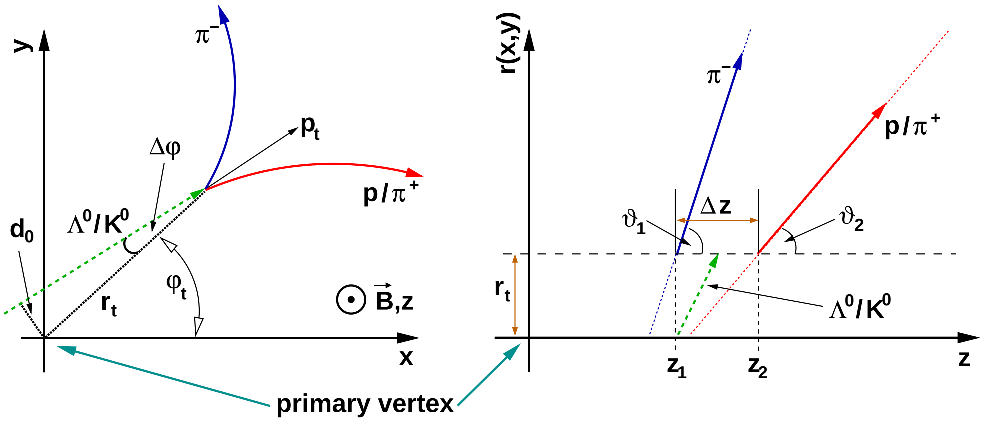


Figure 5.4: Schematic view of the K_S^0/Λ^0 reconstruction in the x-y and r-z plane. Cut quantities used for the selection of the candidates are illustrated in the plots and explained in the text.

where E_{tot} is its total energy and E_i and $\vec{p_i}$ $(i = 1{,}2)$ are the energy and the momentum of the two decay particles.

Afterwards, additional cuts are applied: An upper threshold is set for the absolute value of the pointing angle $|\Delta\phi|$ to reject particles not pointing towards the primary vertex. A cut on $|d_0|$ is applied to constrain the origin of the strange particle to the primary vertex. For the cascade reconstruction (see section 5.6) it is possible to apply a lower cut threshold to $|\Delta\phi|$ and $|d_0|$ in order to select secondary Λ^0.

cut		**particle**	
		K_S^0	Λ^0
r_t	[cm]	$0.7 < \cdots < 25.$	$2. < \cdots < 30.5$
$\|\Delta z\|$	[cm]	$< 60.$	$< 60.$
$\|d_0\|$	[cm]	$0. < \cdots < 2.0$	$0. < \cdots < 0.5$
$\|\Delta\phi\|$	[°]	$0. < \cdots < 18.$	$0. < \cdots < 4.5$
p_t	[GeV/c]	—	—
$\|z_0\|$	[cm]	—	—
m_{inv}	[GeV/c^2]	$0.25 < \cdots < 0.8$	$1.0 < \cdots < 1.5$

Table 5.3: Preselection cuts used for the reconstruction of K_S^0 and Λ^0. Thresholds on p_t or $|z_0|$ are applied at a later stage of the analysis.

After the position of the secondary vertex has been determined, a refitting procedure can be applied to the decay particle tracks in order to improve the

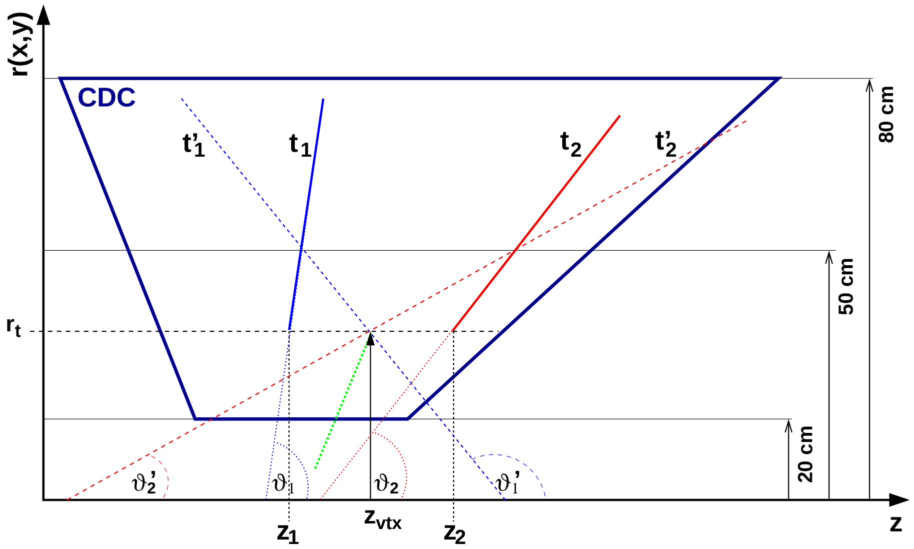


Figure 5.5: Illustration of the track refitting procedure. The dark blue line indicates the longitudinal profile of a CDC sector. The dotted green line denotes the strange particle, the blue and red lines indicate the tracks of the decay particles. The dashed lines mark the refitted tracks.

momentum resolution. A graphical illustration of the procedure is given in figure 5.5. The original tracks t_1 and t_2 of the decay particles have polar angles ϑ_1 and ϑ_2, respectively. At the transverse position (r_t) of the secondary vertex, their longitudinal positions are z_1 and z_2. The refitted tracks t'_1 and t'_2 also have to intersect at the longitudinal coordinate (z_{vtx}) of the secondary vertex which is defined as the mean value of z_1 and z_2. In addition, the track positions remain fixed at a radial distance of 50 cm which corresponds to the middle of the CDC in radial direction. The new polar angles ϑ'_1 and ϑ'_2 are then used to re-calculate the 4-momenta of the decay particles. Thus, the resolution of the strange particle momentum is improved by about 20–25 % which leads to a more narrow peak in the invariant mass spectrum (see section 5.5).

5.4 Combinatorial Background

Strange particle candidates are searched by applying the reconstruction procedure to all possible pairs of decay particles in an event. Therefore, pairs of tracks with one or both particles not coming from the decay of a strange particle can contribute to the invariant mass spectrum.

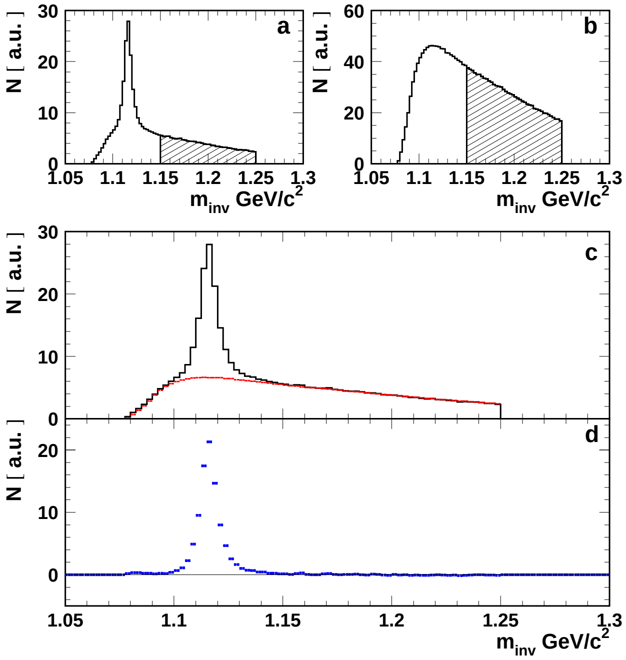


Figure 5.6: Example of the background subtraction procedure for the Λ^0. Pictures *a* and *b* show the combinatorics and the mixed-events background, respectively. The ratio of hatched areas is used to determine the normalization factor for the background subtraction. Combinatorics and scaled mixed-events background (red) are shown in picture *c*. Picture *d* contains the corrected invariant mass spectrum.

An example of the invariant mass spectrum for the Λ^0 is given in panel *a* of figure 5.6. A peak containing the Λ^0 candidates is clearly visible at the nominal mass of 1.116 GeV/c^2 but in addition there is an underlying background ranging from about 1.08 GeV/c^2 (the sum of π^- and proton mass) to the upper end of the spectrum. This background, referred to as *combinatorial background*, results from π^- and proton pairs passing the reconstruction cuts although they are not correlated by a strange particle decay. Due to the protons and pions produced in a central heavy-ion collision there are numerous combinations which can generate fake strange particle candidates in the reconstruction procedure.

For the derivation of the correct number of real strange particle candidates, a method is needed which allows the determination and subtraction of the combinatorial background. In this work, the background is determined by the *event-mixing* method [98, 99]. This is done by combining decay particles coming from two different events and applying the reconstruction procedure. The two particles cannot be correlated by the decay of a strange particle. Therfore, the corresponding invariant mass spectrum will not show the mass peak of the strange particle. Panel *b* of figure 5.6 shows the *mixed-events* background determined by this method.

To make sure that only the correlation of interest is destroyed, the two different events need to have similar properties like orientation of the reaction plane, collision centrality and valid decay particle candidates or strange particle candidates. Therefore, all events are rotated into the reaction plane before analysis and events with valid particle candidates are put into a database according to their Plastic Wall multiplicity. If they are needed for mixing, they are retrieved from this database and they are discarded after being used 20 times. In this analysis, events which are suitable for mixing need to have at least one valid candidate of each decay particle. In order to accumulate enough statistics for the mixed-events background, eight event-mixing reconstructions are performed per real strange particle reconstruction in the case of K^0 and Λ^0 compared to 15 event-mixing cycles for the cascade reconstruction.

The background correction of the measured invariant mass spectrum (figure 5.6, panel*a*) is performed by integrating the spectrum over a range where only a combinatorial background is expected (hatched area). The same range is integrated in the mass spectrum of the mixed-events background (*b*). The ratio of both integrals yields the factor which is used to normalize the mixed-events background spectrum (red histogram in panel *c*) before it is subtracted from the measured spectrum. In panel *d* the background corrected mass spectrum is shown. Only the peak at the nominal mass of the Λ^0 remains while the combinatorial background has vanished. This demonstrates that the event mixing method describes the combinatorial background in the measured spectrum reasonably well.

Spectra of other quantities like the transverse mass or the flow coefficient (cf. chapters 6 and 7) are also corrected with this method. For each of those spectra, the corresponding mixed-events spectrum and the invariant mass spectra of measured data and event-mixing are generated using the same cuts[1]. The normalization factor is determined as before and is afterwards used for subtracting the actual spectra.

5.5 Mass and Phase Space Distributions

After subtracting the background from the measured invariant mass spectra it is possible to determine physical quantities like widths and masses of the strange particle candidates. A better resolution can be achieved by using the refitting method described in section 5.3. This is illustrated in figure 5.7 by comparing background-corrected invariant mass spectra of the Λ^0 generated with and without refitting the tracks. While the peak position agrees for both cases, the width σ and the signal-to-background ratio (within $\pm 2\sigma$) have

---

[1] For the spectra of other quantities than the invariant mass, an additional cut on that mass is applied in order to select the particle candidates.

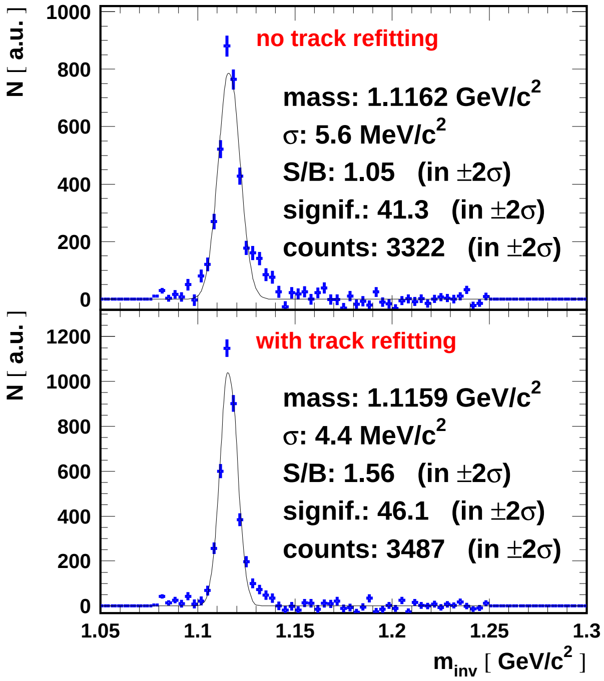


Figure 5.7: Effect of the track refitting procedure. The upper panel shows a Λ^0 invariant mass spectrum generated without using the track refitting, the lower panel shows the same data with track refitting. The peaks were fitted by a Gaussian to extract the mass, width, signal-to-background ratio, significance and the number of candidates.

improved substantially. The width has decreased by more than 20 % from 5.6 MeV/c^2 to 4.4 MeV/c^2 and the signal-to-background ratio has increased from 1.05 to 1.56. As a consequence the significance

$$signif. = \frac{S}{\sqrt{S+B}} \tag{5.2}$$

(S: signal counts, B: background counts) of the invariant mass peak has improved slightly and in addition the number of reconstructed Λ^0 candidates has increased by 5 %.

The final invariant mass spectra of K^0 and Λ^0 are presented in figure 5.8. The peak in the background-corrected K^0_S spectrum is located at 503 MeV/c^2 which is only slightly larger than its nominal mass. The width of 15.8 MeV/c^2 is due to the longitudinal resolution of the CDC, its intrinsic width is orders of magnitude smaller. Within the $\pm 2\sigma$ interval around the peak center the signal-to-background ratio is about 1.0 and the significance is 106 for a total statistics of more than 22000 K^0_S candidates. Set '1' of the final K^0 cuts was used here.

The corresponding mass spectrum of the Λ^0 candidates shows a peak with a smaller width compared to the K^0_S. Its position is at 1116 MeV/c^2

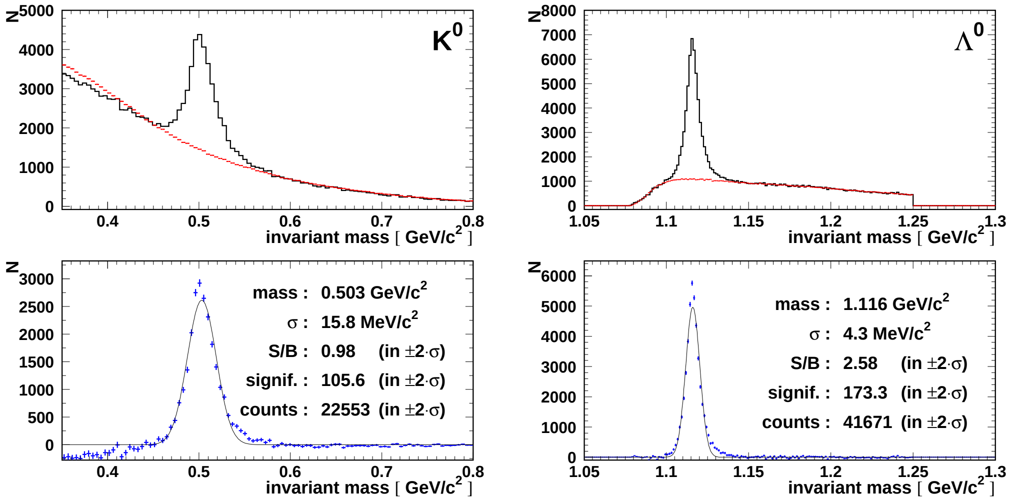


Figure 5.8: Invariant mass spectra of reconstructed K^0_S (left) and Λ^0 (right). The upper panels show the combinatorics (black) and the normalized mixed-events background (red), the lower panels show the background subtracted mass peaks.

which is very close to its nominal mass. The width of 4.3 MeV/c^2 is given by the CDC momentum resolution. The smaller width compared to the K^0 is due to the reduced mass difference between mother particle and decay products. Therefore, the particle momenta play a minor role in the invariant mass determination. Within the $\pm 2\sigma$ interval approximately 41000 Λ^0 candidates with a signal-to-background ratio of 2.6 and a significance of 173 are measured. Cut set '1' was applied (see appendix E) here.

	cut	mass [GeV/c^2]	width [MeV/c^2]	S/B	signif.	N$_{part}$
K^0_S	**1**	0.5030	15.8	1.07	107	22133
	2	0.5027	16.0	1.75	116	21106
Λ^0	**1**	1.1161	4.3	2.58	173	41671
	2	1.1162	4.7	1.73	185	53880
	3	1.1161	4.7	5.32	168	33370

Table 5.4: Properties of the reconstructed K^0 and Λ^0 candidates under different cut sets. The sets used in this work are printed in bold letters.

Properties of the reconstructed K^0 and Λ^0 candidates after centrality selection cut are given in table 5.4 for all available cut sets. Positions and widths of the peaks are in good agreement for all sets whereas signal-to-

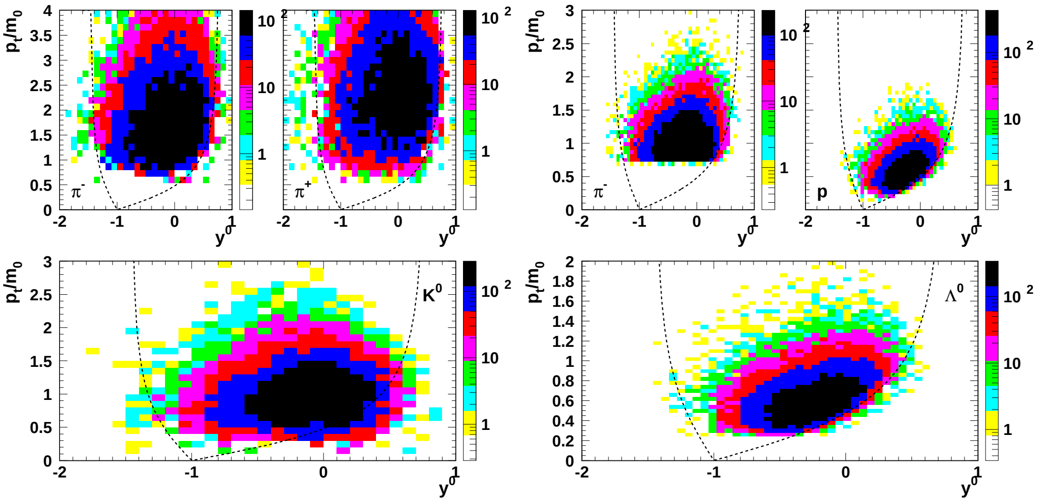


Figure 5.9: Phase space distributions of K^0 (left) and Λ^0 (right) candidates (lower panels) and their decay particles (π^-, π^+ and proton) (upper panels). The dashed lines indicate the polar angular coverage of the CDC.

background ratio, significance and the number of reconstructed particle candidates differ substantially.

Figure 5.9 presents the background-corrected phase space distributions of K^0_S, Λ^0 and their respective decay particles. The normalized transverse momentum p_t/m_0 is plotted versus the scaled rapidity y^0 (see appendix C). These distributions are used to verify the reconstruction procedure and to investigate the reconstruction efficiency (cf. chapter 6). The geometrical acceptance of the CDC is indicated by the dashed lines. While the decay particles have to come from within this region of acceptance, the strange particle candidates can also lie outside, depending on their decay kinematics.

The K^0 distribution shows that a substantial fraction of candidates comes from the rapidity range $0 < y^0 < 0.5$ outside the geometrical acceptance. This is possible because the decay particles lie within the acceptance due to the momenta they gain in the decay. The sharp cutoff at the lower end of the decay particle distributions reflects the minimum p_t threshold (> 90 MeV/c) used in the analysis. Particles which are outside the CDC acceptance in those plots are a result of the momentum resolution.

In comparison to K^0 the phase space distribution of the Λ^0 is concentrated at mid-rapidity and only a few candidates do not come from the region of geometrical acceptance. Due to the smaller excess energy in the decay the decay particles show a less pronounced spread in the p_t/m_0 direction. The p_t threshold of > 100 MeV/c is visible in the π^- distribution whereas the

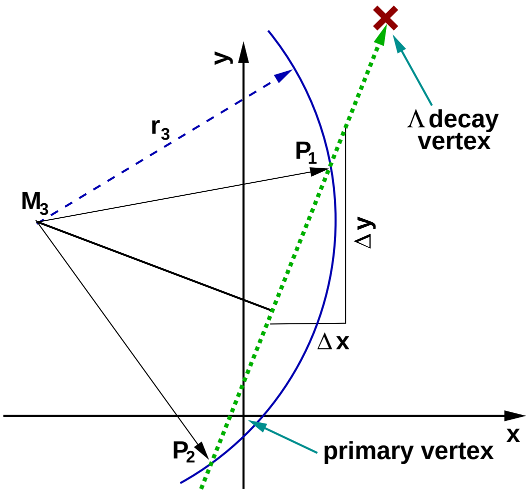


Figure 5.10: Determination of the secondary vertex for Ξ^- and $\Sigma^-(1385)$. The solid blue line and the dotted green line indicate the tracks of the π^- and the Λ^0, respectively. The geometrical reconstruction is explained in the text.

protons are not affected by this cut.

5.6 Reconstruction Method for Ξ^- and $\Sigma^-(1385)$

Cascade particles like Ξ^- and $\Sigma^-(1385)$ have to be reconstructed in two steps. In the first step the Λ^0 is reconstructed with the method explained in section 5.3. In a next step the Λ^0 is combined with another π^- in order to reconstruct the cascade particle.

Figure 5.10 illustrates a sketch of the geometrical reconstruction procedure for a cascade decay is illustrated. The idea behind this is to find intersection points of a π^- track and the reconstructed Λ^0 in the transverse plane. The arc described by the π^- is defined by the center M_3 and the radius r_3, the flight path of the Λ^0 is given by its decay vertex and the components of its transverse momentum (Δx, Δy). The refitting procedure explained in section 5.3 is also applied to this reconstruction. Here, the z position of the Ξ^- decay vertex is solely defined by the Λ^0. Hence, only the π^- track is refitted.

The two panels in figure 5.11 illustrate important cut quantities in the x-y and r-z plane. While r_t, ϕ_t, $\Delta\phi$ and Δz are determined in the same way as for the K^0 or Λ^0 reconstruction, the calculation of d_0 is different because of the negative charge of the mother particle[2].

The preselection cuts used in the reconstruction of Ξ^- and $\Sigma^-(1385)$ are given in table 5.5. Due to the very short lifetime of the $\Sigma^-(1385)$ the cut on r_t can be restricted to a range of 0 to 2 cm. Under this cut condition it is

---

[2] This is of importance only for the Ξ^- because all $\Sigma^-(1385)$ decay directly in the collision zone.

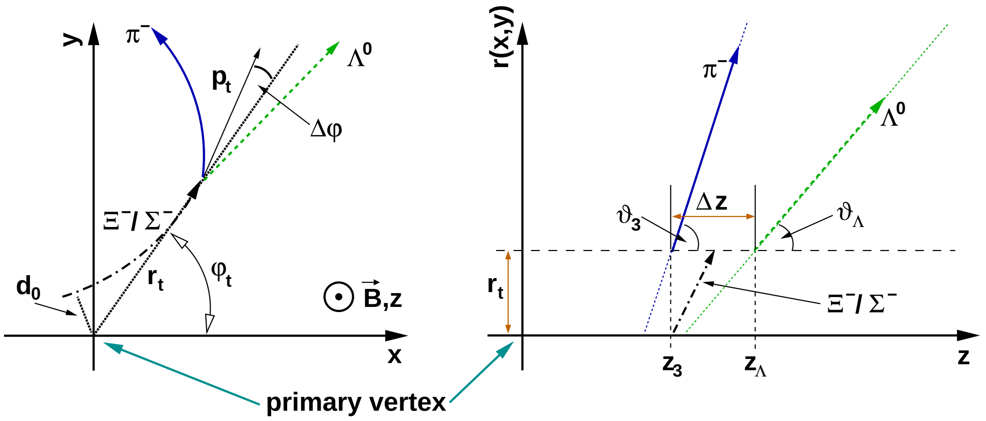


Figure 5.11: Schematic view of the $\Xi^-/\Sigma^-(1385)$ reconstruction in the x-y and r-z plane. The quantities used for the selection of the candidates are illustrated in the plots.

cut		**particle**	
		$\boldsymbol{\Xi^-}$	$\boldsymbol{\Sigma^-(1385)}$
r_t	[cm]	$0.25 < \cdots < 15.$	$0. < \cdots < 2.$
$\lvert\Delta z\rvert$	[cm]	$< 50.$	$< 100.$
$\lvert d_0\rvert$	[cm]	$0. < \cdots < 0.4$	$0. < \cdots < 0.5$
$\lvert\Delta\phi\rvert$	[°]	$0. < \cdots < 15.$	$0. < \cdots < 180.$
p_t	[GeV/c]	—	—
$\lvert z_0\rvert$	[cm]	—	—
m_{inv}	[GeV/c^2]	$1.100 < \cdots < 1.750$	$1.100 < \cdots < 1.750$

Table 5.5: Preselection cuts used for the reconstruction of Ξ^- and $\Sigma^-(1385)$. Thresholds on p_t and $|z_0|$ are applied later in the analysis.

verified that the 1st π^- directly originates from the primary vertex position. Therefore, the quantity $\Delta\phi$ is not well defined and the corresponding cut is opened to a range of 0° to 180°.

Chapter 6

Acceptance and Efficiency Corrections

In this chapter we discuss corrections which are due to the geometrical acceptance and to other inefficiencies of the CDC. The geometrical acceptance is given by the active volume of the detector and is about 70 % of the full solid angle 4π. The detector efficiency is determined by factors like loss of particles in the material between target and active volume, two-track resolution, track finding inefficiencies and particle identification capabilities. In order to extract quantitative results from the measured data it is necessary to correct all losses due to acceptance and efficiency effects. For a complex detector like the FOPI this correction has to be determined by an extensive simulation and modeling of all detector parts including all relevant physics effects.

The chapter will briefly introduce the principle of the detector simulation. Afterwards the consistency between simulation and experimental data is investigated by comparing distributions of certain cut quantities for the particle selection and reconstruction. Finally, the reconstruction efficiency for K^0 and Λ^0 is presented and efficiency corrections for the particle yields are derived.

6.1 Simulation of the Detector

The simulation of the detector response is realized with the GEANT package (version 3.21) from CERN [100]. It allows to simulate a three-dimensional model of the detector, taking into account all important materials. Particles are propagated through the detector, undergoing all relevant physics processes like electromagnetic and hadronic interactions, multiple scatter-

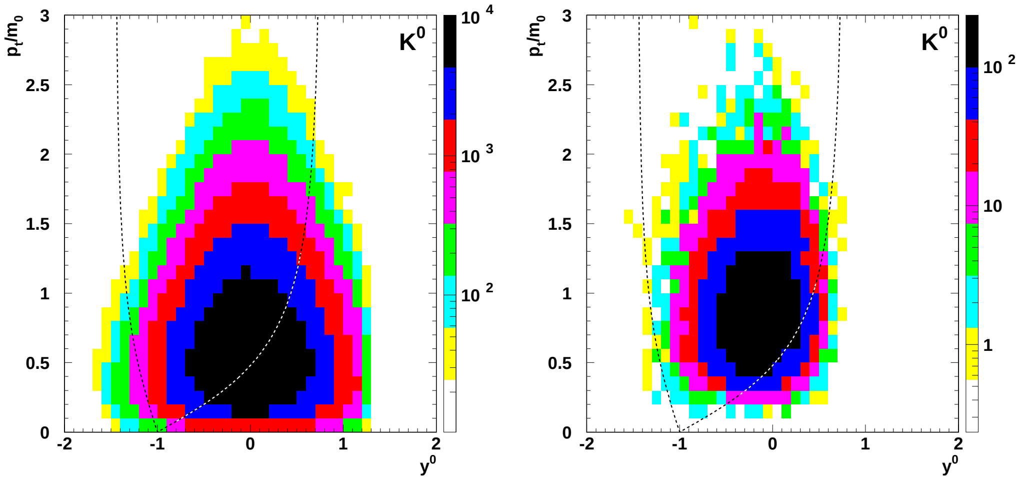


Figure 6.1: Phase space distributions of the thermally generated (left) and of the reconstructed (right) K_S^0 in the simulation. The dashed lines indicate the CDC acceptance defined by the polar angle.

ing and decay. The response of the various sub-detectors to the traversing particles is modeled: In particular, this applies for the electron drift in the drift chamber gas, signal creation and propagation, FADC digitization, zero suppression and data reduction. In the end, output data equivalent to that of a real experiment is created. Thus, simulated and measured data can be treated with the same analysis program. All sub-detectors (Barrel, CDC, Helitron, Plastic Wall and Zero Degree) and the magnet are included. The magnetic field is parameterized for the simulation. A schematic drawing of the detector model coming from the FOPI GEANT environment was already presented in figure 2.1.

In order to simulate heavy-ion collisions, particle distributions are needed which describe the global features like particle multiplicity and phase space distributions of a real event. In our case, the IQMD model [51] (see also appendix A) is used as an event generator[1].

Strange particles under consideration are generated separately and then *embedded* in the simulated events. Depending on the analysis this can be one or more particles per event. For example, Λ^0 and K^0 are embedded in the generated events together. This has to be done in order to account for the

---

[1] Given the fact that IQMD generates strange particles according to their physical production probability, the amount of K^0 and Λ^0 within the sample of simulated events is very small (about 0.1 per event). In order to increase the amount of strange particles in the events and to limit the computing time, strange particles are embedded in those events later on.

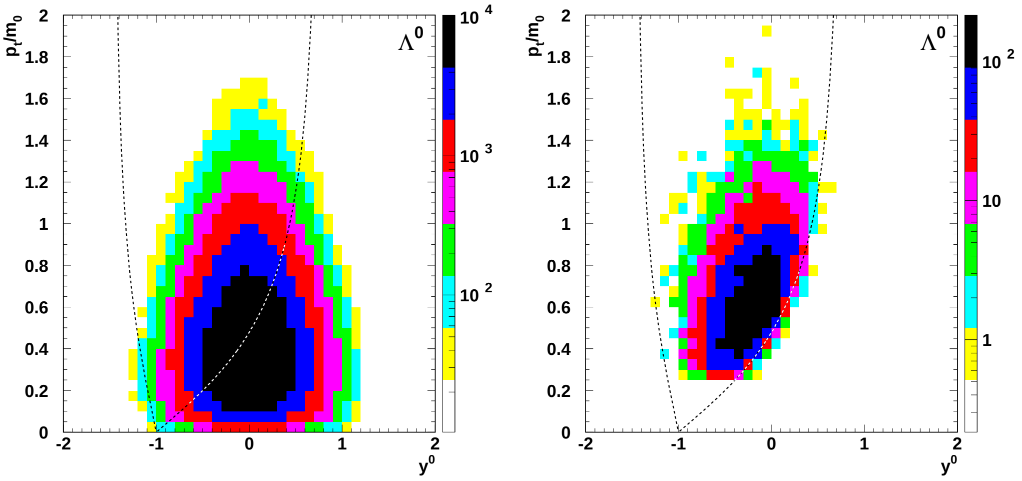


Figure 6.2: Phase space distributions of the thermally generated (left) and of the reconstructed (right) Λ^0_S in the simulation. The dashed lines indicate the CDC acceptance defined by the polar angle.

fact that due to their associated production the π^- coming from the decay of one of the particles could contribute to the signal of the other particle when analyzing measured data. The momentum distribution is generated with the help of the Siemens-Rasmussen formula [101] (cf. equation 6.1) which describes a system expanding with a radial velocity of β and a thermal momentum distribution according to a temperature T.

$$\frac{d\sigma}{dp} = p^2 \cdot \exp\left(-\gamma\frac{E}{T}\right) \cdot \left[\left(\gamma + \frac{T}{E}\right)\frac{\sinh\alpha}{\alpha} - \frac{T}{E}\cosh\alpha\right] \tag{6.1}$$

$$E = \sqrt{m_0^2 + p^2}, \qquad \gamma = \frac{1}{\sqrt{1-\beta^2}}, \qquad \alpha = \frac{\gamma\beta p}{T}$$

Here p and E are the total momentum and total energy in the center-of-mass system. The values for the parameters are $\beta = 0.3$ and $T = 90$ MeV which are close to those values determined for π^-, protons and deuterons in the Ni+Ni system at 1.93 AGeV [102]. The branching ratios for the relevant decay channels (see table 5.1) are set to 100 % in GEANT to enhance the available statistics. In our case, about 3.5 millions of simulated events with embedded K^0_S and Λ^0 and several ten thousands of events with embedded Ξ^- or $\Sigma^-(1385)$ are available for the determination of the efficiency correction.

When all the particles of an event have been propagated through the detector, the response of the different sub-detectors is calculated. For the CDC, the energy loss in each drift cell is converted into the charge induced on the

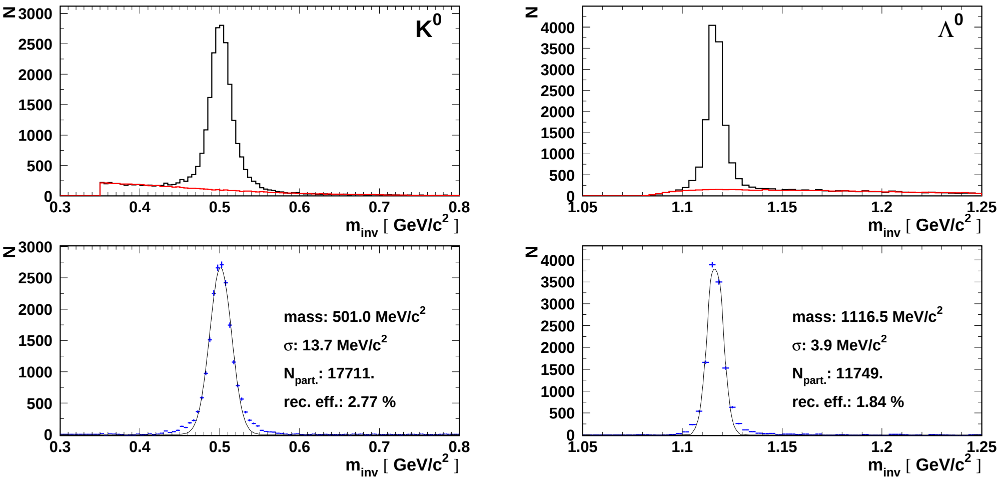


Figure 6.3: Invariant mass spectra of reconstructed K^0_S (left) and Λ^0 (right) from the simulation.

corresponding wire. Taking into account resolution effects and other corrections, the collected charge is then translated into a signal which is equivalent to the output of the FADCs in the experiment. Subsequently the signals are processed by an offline version of the data reduction algorithm which runs on the DSPs during an experiment (cf. section 3.4 of chapter 3). The output of the full simulation chain can therefore be used as an input for the standard offline software which is used for calibration, tracking and particle identification within FOPI. In addition to the usual data, the final output of the simulation also contains information about the input events like vertex position, impact parameter, reaction plane orientation, particle species and particle momenta.

The phase space distributions of thermally generated K^0_S and Λ^0 which were embedded in the simulation are shown in the left pictures of figure 6.1 and 6.2, respectively. Due to their lower mass, the K^0_S can populate a larger region of the available phase space. The right pictures of both figures present the background corrected phase space distributions of particle candidates reconstructed from the simulated data. These distributions are in agreement with those from the experiment given in figure 5.9. To select the same fraction of the total cross section as for the measured data, a threshold of PMUL >26 was applied to the simulated events.

The overall reconstruction efficiency is determined by comparing the number of reconstructed candidates to the number of simulated events being processed in the analysis. Figure 6.3 shows the invariant mass spectra of reconstructed K^0_S and Λ^0 candidates from the simulation after centrality selection. Position and width of the peaks in the spectra are extracted from a Gaussian fit. While for both particles the mean mass agrees with the experiment (see fig. 5.8), the peak widths are slightly smaller. This indicates that the resolution of track properties in the simulation is somewhat better than in the experiment. After analyzing the simulated events more than 17000 K^0 candidates are found. Due to the branching ratio of 100 % the global reconstruction efficiency is approximately 2.8 %. When searching the same data sample for Λ^0 candidates more than 11000 candidates remain. This corresponds to a reconstruction efficiency of 1.8 %. The difference between the efficiencies can be understood from the decay kinematics of the particles. Compared to those of the Λ^0 the decay particles of the K^0 have a higher probability to be within the CDC acceptance even if this is not the case for the mother particle. This is visible in the phase space distributions of simulation and experiment.

6.2 Comparison of Cut Quantities

In order to get a consistent efficiency correction one has to ensure that particle properties and detector response are described reasonably well by the simulation. For this purpose, the distributions of cut quantities which were used for the strange particle reconstruction are compared between simulated and measured data. The comparison is restricted to the transverse momentum p_t, the closest distances to the primary vertex d_0 and z_0, the mass m_{CDC} determined from CDC information and the hit multiplicity per track.

The spectra for simulated and measured data are generated by first applying the centrality cut (PMUL >26) and then selecting the mother particle candidates from an interval of $\pm\ 2{\cdot}\sigma$ around the peak in the invariant mass spectrum. For these candidates and their decay particles the background corrected spectra are generated and compared. Due to the different number of events in simulation and measured data the spectra are afterwards normalized to their particular number of entries to allow a direct comparison of the shape of the distribution.

6.2.1 K^0_S Cut Spectra

Figure 6.4 shows the distributions for cut quantities of the K^0_S decay particles π^- (left column) and π^+ (right column). The red histograms represent the distributions from the simulation, the black histograms those from the measured data. An excellent agreement between simulation and data is found

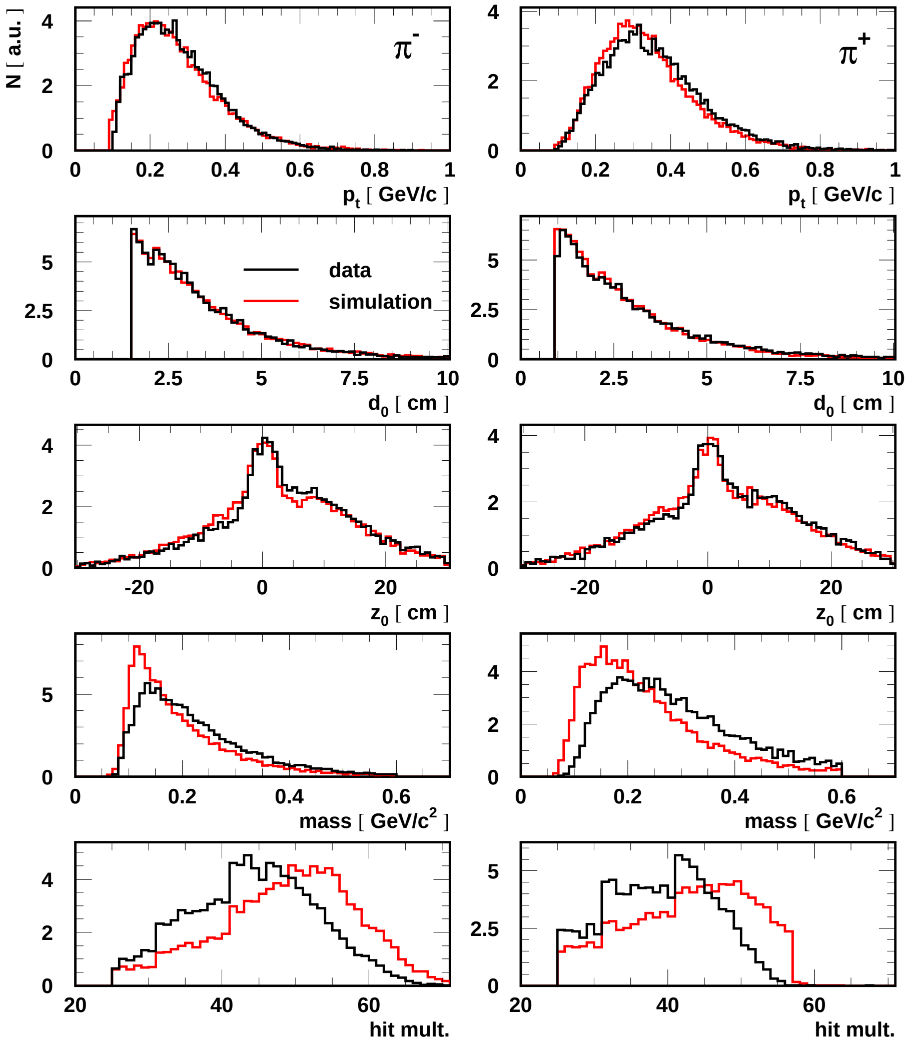


Figure 6.4: Spectra of the cut quantities for the K_S^0 decay particles π^- (left) and π^+ (right). The transverse momentum p_t, the closest distances to the primary vertex d_0 and z_0, the particle mass and the track's hit multiplicity are plotted.

for the transverse momentum p_t distribution and for those of the closest distances to the primary vertex (d_0, z_0).

A difference is found between the CDC mass spectra of simulated and measured data. The particle mass within the simulation is shifted towards lower values as in the data and this effect is more pronounced for the π^+ than for the π^-. This effect is due to an imperfect description of the energy loss

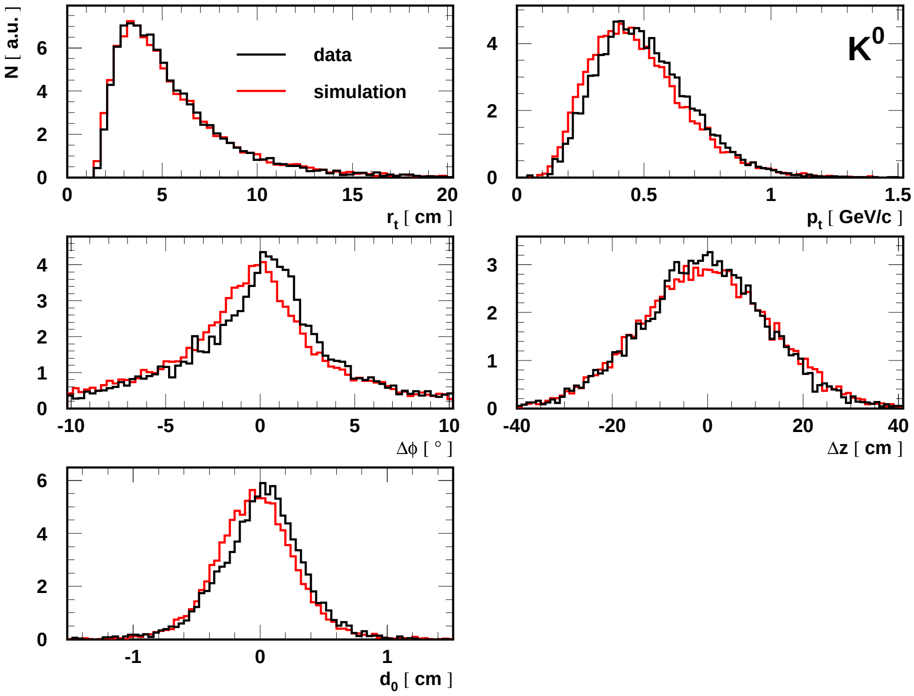


Figure 6.5: Spectra of the cut quantities for the K^0_S. The transverse distance to the secondary vertex r_t, the transverse momentum p_t, the pointing angle Δ_ϕ, the z distance of the π^- and π^+ tracks Δ_z and the closest distance to the primary vertex d_0 are plotted.

dE/dx which is used to determine the particle mass in the CDC. In order to guarantee that the efficiency correction will not be biased, a wide mass window is used in the analysis. The disagreement which is found between the hit multiplicity spectra of simulated and measured data is a consequence of track finding inefficiencies in the experiment. Noise signals which are picked up by the sense wires (cf. figure 3.5) are not modeled in the simulation. Thus, the signals found on the sense wires are cleaner than in the experiment, the data reduction algorithm finds more hits and the tracking routine will find longer tracks on average. Lowering the hit multiplicity threshold for the π^- will slightly increase the ratio of particles found in the measured data and the simulation. This could cause a systematically higher yield after applying the efficiency correction and has to be included into the systematic error. For the π^+ this effect is less severe due to the already lower hit multiplicity threshold.

The spectra for the cut quantities for the K^0_S are presented in figure 6.5.

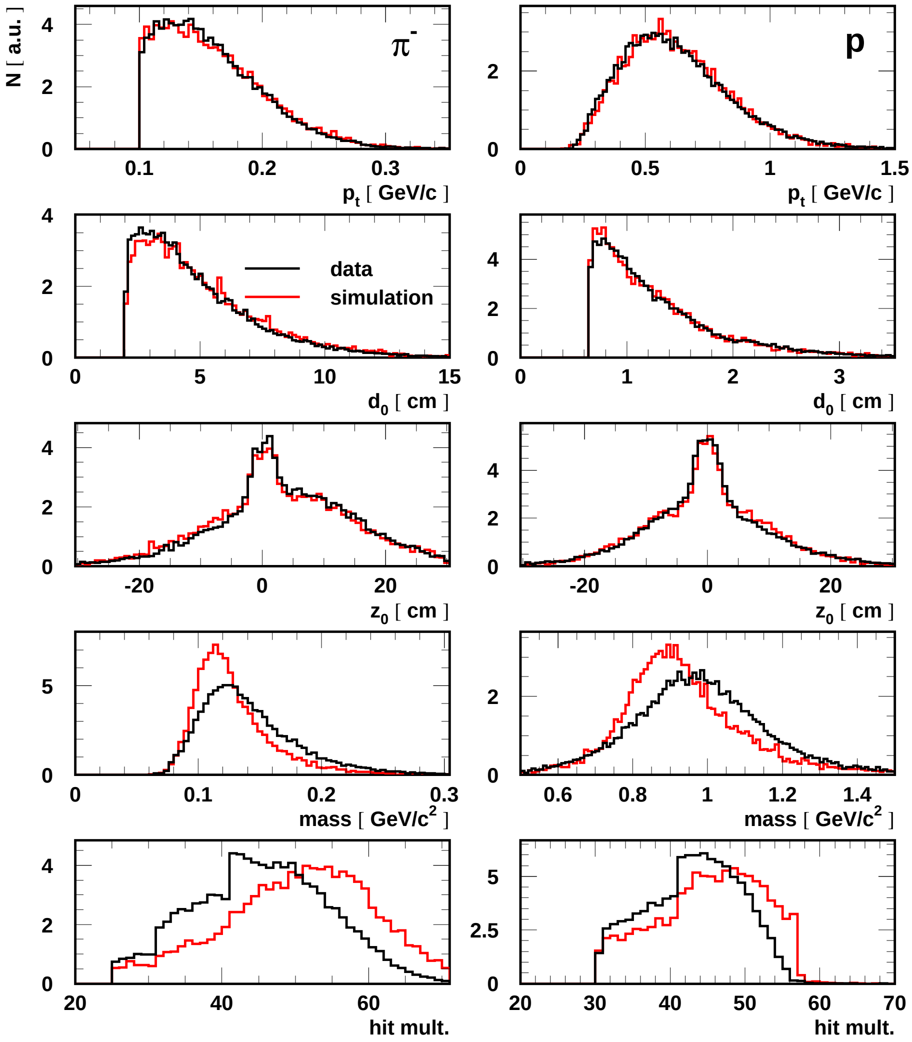


Figure 6.6: Spectra of the cut quantities for the Λ^0 decay particles π^- (left) and p (right). The transverse momentum p_t, the closest distances to the primary vertex d_0 and z_0, the particle mass and the track's hit multiplicity are plotted.

This comparison shows that the spectra of transverse distance r_t, transverse momentum p_t and the z distance of the decay particles π^- and π^+ nicely agree between simulation and experiment. Minor differences are found for the distributions of the pointing angle $\Delta\phi$ and the closest distance to the vertex d_0 which are linked by the relation

$$d_0 = r_t \cdot \sin(\Delta\phi) \ . \tag{6.2}$$

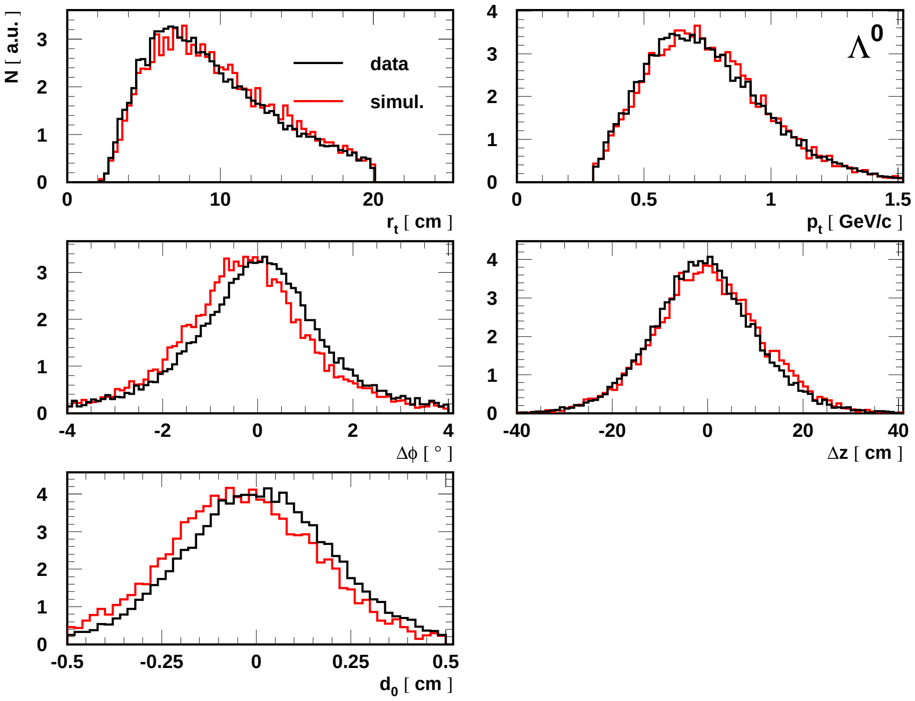


Figure 6.7: Spectra of the cut quantities for the Λ^0. The transverse distance to the secondary vertex r_t, the transverse momentum p_t, the pointing angle Δ_ϕ, the z distance of the π^- and π^+ tracks Δ_z and the closest distance to the primary vertex d_0 are plotted.

These differences are not critical because no strong constraints are applied in the reconstruction procedure.

6.2.2 Λ^0 Cut Spectra

Distributions of the cut quantities for the Λ^0 decay particles π^- (left column) and p (right column) are plotted in figure 6.6. As for the K^0, a good agreement is found for the p_t, d_0 and z_0 distributions. The distributions of m_{CDC} exhibit the same qualitative behaviour like those of the K_S^0 decay particles. Again, the mass window for both decay particles is open. Therfore, differences between simulation and data will not affect the final results. The average hit multiplicity is once more significantly higher for tracks in the simulation due to the reasons discussed before. However, the cut thresholds which were applied (> 24 for the π^-, > 29 for the proton) are non-critical because they reject approximately the same number of candidates from mea-

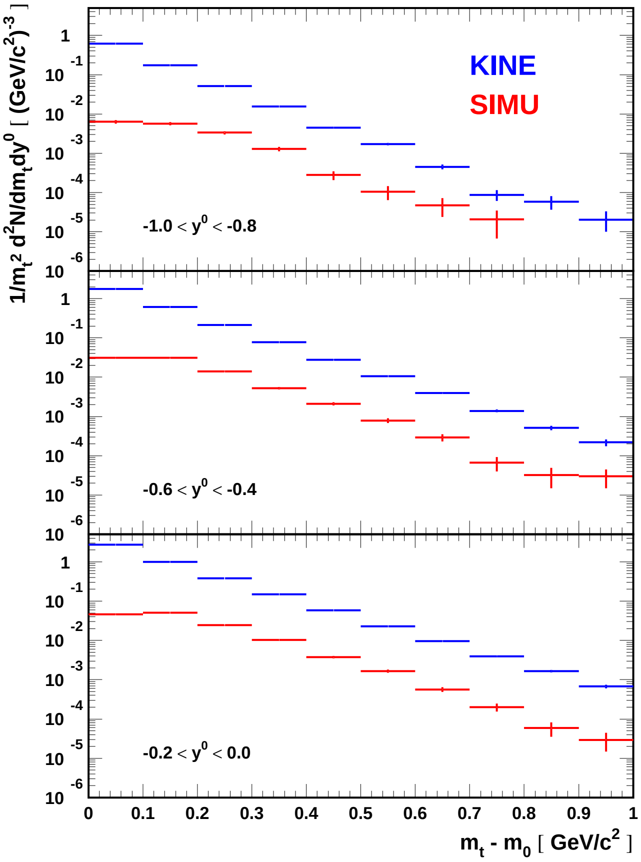


Figure 6.8: Transverse mass spectra of embedded thermal K_S^0 (KINE) and reconstructed K_S^0 (SIMU) in the simulation. The spectra are plotted for target rapidity (-1.0 $< y^0 <$ 0.8, upper plot), mid-rapidity (-0.2 $< y^0 <$ 0.0, lower plot) and for another bin of rapidity in-between the two other bins (-0.6 $< y^0 <$ -0.4, middle plot).

sured data and simulation. Thus the efficiency correction is not affected. Cutting stronger on this quantity would decrease the corrected Λ^0 yield, though.

Regarding the Λ^0 distributions in figure 6.7 one finds a very good overall agreement. A slight asymmetry is visible in the distribution of $\Delta\phi$ which also propagates into that of the d_0. In the case of the Λ^0 both quantities are not subject to strong cuts either so that an efficiency correction can be determined.

6.3 Determining the Efficiency Correction

The reconstruction efficiency for K_S^0 and Λ^0 is determined as a function of the transverse mass m_t and the scaled rapidity y^0. The phase space is divided into cells in longitudinal $(m_t - m_0)$ and transversal (y^0) direction and the reconstruction efficiency is computed separately for each cell. The inverse of

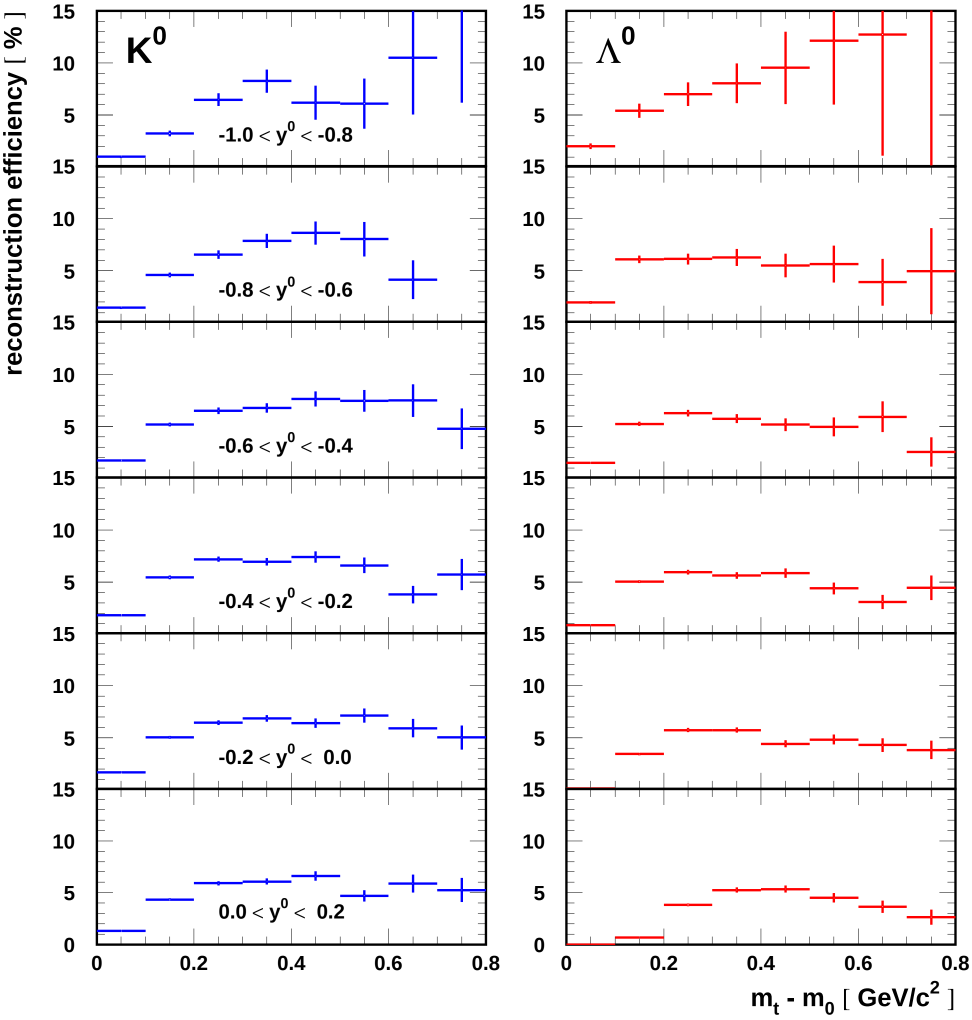


Figure 6.9: Reconstruction efficiency of the K^0_S (left) the Λ^0 (right) as function of the transverse mass $m_t - m_0$ and the scaled rapidity y^0 interval.

the efficiency is then used as a correction factor for the measured data.

Examples of transverse mass spectra for reconstructed K^0 (SIMU) and embedded thermal K^0 (KINE) are given in figure 6.8 for three bins (target rapidity, mid-rapidity and in-between) in y^0. The corresponding spectra of the Λ^0 are not provided here. Due to the fact that the embedded particles are thermally generated, a Boltzmann-like behaviour (cf. equation 7.1 in chapter 7) is visible in their spectra. At high transverse masses this behaviour is also found for the reconstructed candidates, whereas it is different at low values of m_t.

The ratio of the transverse mass spectra for reconstructed particles from simulated events (SIMU) and for all particles embedded into the simulated events (KINE) represents the reconstruction efficiency for a given bin of y^0. The result for K_S^0 and Λ^0 is presented in figure 6.9. Due to the transverse momentum thresholds for the decay particles the efficiency is low at small transverse masses. Around mid-rapidity, no Λ^0 candidates are reconstructed in the lowest m_t bin as a consequence of the geometrical acceptance and the decay kinematics. At higher m_t values the reconstruction efficiency is approximately constant at values of 7 % for the K_S^0 and about 5 % for the Λ^0. In addition, the efficiencies are roughly independent of y^0. This means that the shape of the input distribution of the embedded particles is not that important and that the efficiency-corrected data are not biased by the correction method.

Chapter 7

K^0 and Λ^0: Results

In the previous chapter the determination and application of the efficiency correction was described. Now, the corrected transverse mass spectra of K^0 and Λ^0 are used to extract an effective temperature and, by integration, a rapidity-dependent yield distribution. Temperature, total yield and width of the distribution can later be compared to already existing data and to model calculations. Furthermore, azimuthal distributions of the produced particles are studied.

Unless stated differently, all results presented in this chapter refer to a most central 350 mb out of the total geometrical cross section of 2.7 b. This selection, which is done by requiring the charged particle multiplicity measured in the Plastic Wall to be larger than 31, corresponds to impact parameters smaller than 3.3 fm. All errors given in the plots and the text are statistical. An estimation of systematic errors is presented in subsection 7.1.4.

7.1 Kinematic Distributions

7.1.1 Transverse Mass Spectra

Transverse mass spectra of K_S^0 and Λ^0 after background and efficiency correction are plotted in figure 7.1. As a consequence of the shifted target position and the high-statistics data sample, rapidities from -1 (target rapidity) to about +0.4 (beyond mid-rapidity) are covered (cf. acceptance plots in figure 5.9 on page 65).This means that essentially the full phasespace is accessible to the detector. The spectra were scaled with factors 10^n, where $n = 0$ for the spectrum at target rapidity (-1.0< y^0 <-0.8) up to $n = 6$ for the spectrum at +0.2< y^0 <+0.4.

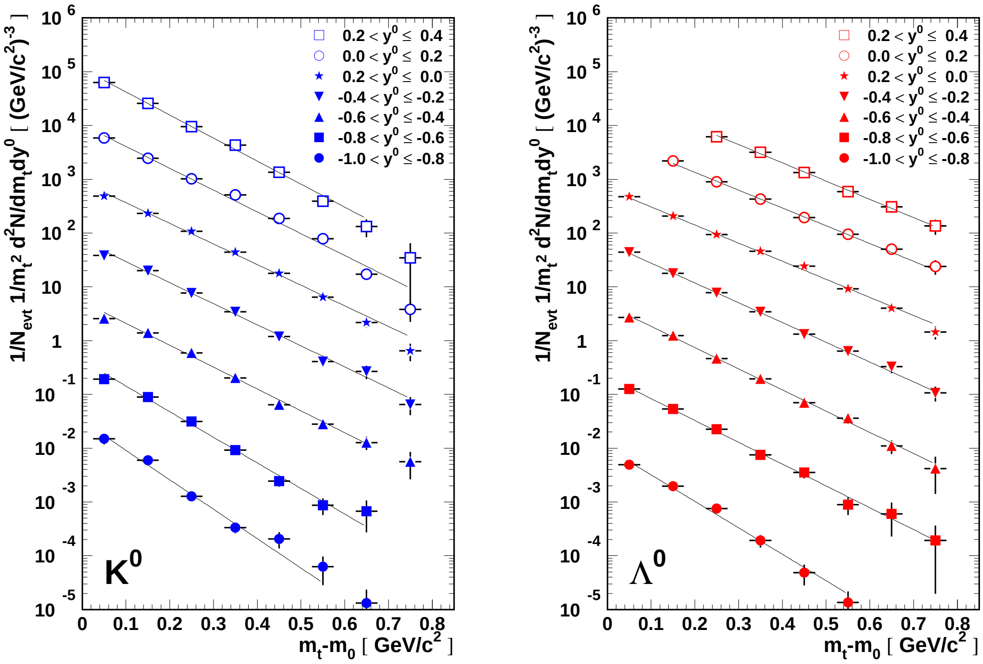


Figure 7.1: Transverse mass spectra of reconstructed K_S^0 and Λ^0. The spectra are plotted for different bins in rapidity within the range -1.0 < y^0 < 0.4. The lines represent fits of a Boltzmann-like function.

Effects of the geometrical limits of the detector and the reconstruction cuts can be seen at very low values of m_t. Due to the lower p_t threshold and the limitation of the forward polar angle to values larger than 23° the lowest m_t bins are missing for projectile rapidities in case of the Λ^0. The situation is much better for the K^0. Its particular decay kinematics lead to an enhancement of the acceptance in the forward region (cf. figure 5.9). For tranverse masses larger than 0.7 statistics become scarce which is reflected by the error bars.

In the absence of collective effects, particle emission in a thermalized system can be described by a Boltzmann-like distribution:

$$\frac{1}{m_t^2}\frac{d^2N}{dm_t dy^0} = A_B(y^0) \cdot \exp\left(-\frac{m_t - m_0}{T_B(y^0)}\right) , \tag{7.1}$$

where both, the parameter A_B and the inverse slope parameter T_B are functions of the scaled rapidity y^0. In a logarithmic representation the exponential decrease will be visible as a straight line. If collective effects are present, the fit delivers an *effective temperature* T_{eff}. Non-thermal contributions to the m_t spectra will cause deviations from the exponential behaviour.

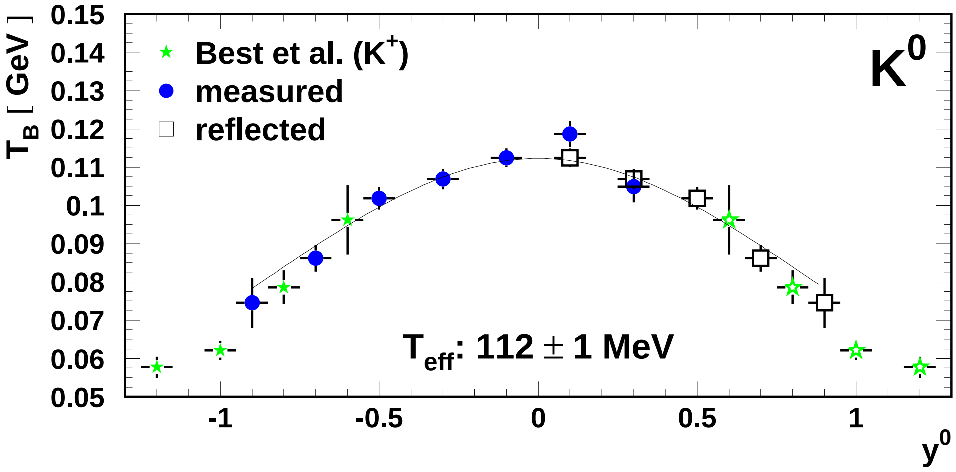


Figure 7.2: Inverse slope parameter distribution of K_S^0 candidates. The blue circles indicate measured data, the hollow black squares are data points reflected at mid-rapidity. The line represents a fit assuming an isotropic thermalized source. The green symbols mark the results of a previous K^+ measurement [50].

In the logarithmic representation the measured spectra exhibit a linear decrease with increasing $m_t - m_0$ and are described reasonably well by a Boltzmann distribution. By fitting equation 7.1 to the corrected spectra one obtains y^0-dependent distributions of the parameters T_B and A_B. In the following sections they are used to determine the effective temperature and the differential yield distribution.

7.1.2 Inverse Slope Parameter

In a thermalized system, the inverse slope parameter T_B yields information on the particle temperature at freeze-out. For an isotropically emitting source, the Boltzmann temperature T_B has a simple dependence on the rapidity:

$$T_B(y^0) = \frac{T_{eff}}{\cosh(y^0)} \tag{7.2}$$

where T_{eff} is the slope parameter at mid-rapidity or the effective temperature.

The slope parameter distribution of the K^0 candidates is presented in figure 7.2. The blue circles mark the measured data, the hollow black squares are data points reflected at mid-rapidity. An effective temperature of 112±1 MeV can be extracted from a fit using equation 7.2. The green

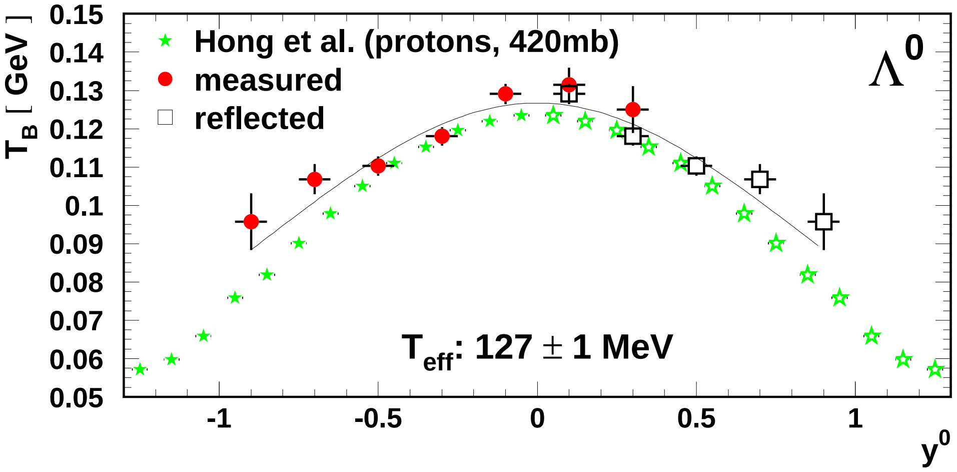


Figure 7.3: Inverse slope parameter distribution of Λ^0 candidates. The red circles indicate measured data, the hollow black squares are data points reflected at mid-rapidity. The line represents a fit assuming an isotropic thermalized source. The green symbols mark the corresponding proton distribution [102].

symbols indicate published K^+ data from a previous experiment [50] studying Ni+Ni collisions at the same beam energy. Due to the fact that the target was in its nominal position and that the charged kaons were measured directly, a different rapidity region was covered at that time. In the overlap region a good agreement between both experiments is found. This suggests that K^0 and K^+ have the same freeze-out conditions.

Figure 7.3 shows the corresponding distribution for reconstructed Λ^0 candidates. Now red circles indicate measured data and hollow black squares again are data points reflected at mid-rapidity. Here, the effective temperature taken from the fit is about 127±1 MeV. Inverse slope parameters of protons [102] in the same system are depicted by the green symbols. A value of $T_{eff} = 125$ MeV was determined. The proton data shown in the plot were measured with a centrality selection of 420 mb. Analyses of more central data samples (100 mb, 250 mb) were also described in that publication but no particular difference to the 420 mb data was found. While the data agree in the mid-rapidity region, the proton distribution drops more rapidly when going to larger absolute values of y^0. In contrast to the Λ^0 which are produced in the collisions, protons are emitted non-isotropically. The lower T_B values around target and projectile rapidities correspond to lower average values of the transverse momentum indicating the presence of cold spectator

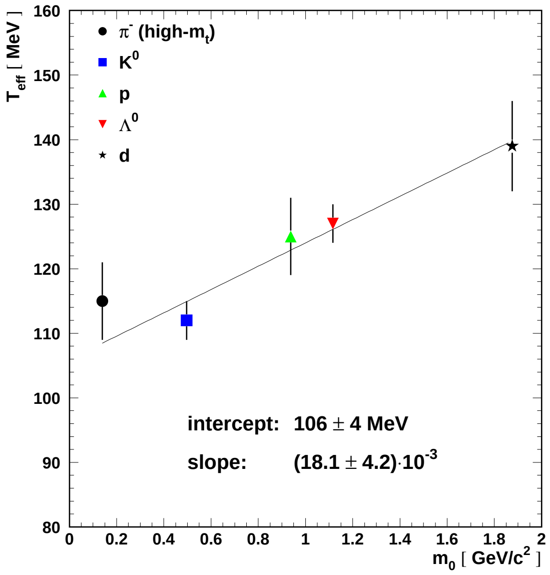


Figure 7.4: Mass dependence of the effective temperature for various particle species. The K^0 and Λ^0 results are from this work, the values for π^-, protons and deuterons were taken from [102]. A linear dependence of T$_{eff}$ on the particle mass can be observed.

matter.

The effective temperatures of Λ^0 and K^0 which were determined in this work and those of π^-, protons and deuterons published in [102] are plotted versus the mass of the particles in figure 7.4. A linear behaviour of T$_{eff}$ on the particle mass is found. By fitting a line, an intercept of 106 ± 4 MeV and a slope of $(18.1 \pm 4.2)\cdot 10^{-3}$ are extracted. Assuming that the mean kinetic energy is proportional to the effective temperature

$$\langle E_{kin} \rangle = \frac{3}{2} T_{eff} \tag{7.3}$$

and decomposing $\langle E_{kin} \rangle$ into one part depending on the temperature T and another part depending on the radial expansion velocity β_r in a non-relativistic approximation [103] one obtains

$$T_{eff} = \frac{2}{3} \cdot \left[\frac{m_0}{2} \langle \beta_r \rangle^2 + \frac{3}{2} T \right] \; . \tag{7.4}$$

The radial expansion velocity derived from the slope in figure 7.4 is $\langle \beta_r \rangle = 0.23$.

7.1.3 Differential Yield Distributions

The differential yield distribution dN/dy^0 is determined by an analytic integration of equation 7.1 using the parameters A_B and T_B taken from the fit. Integrating from $m_t = 0$ to infinity, one obtains

$$\left. \frac{dN}{dy^0} \right|_{y_i^0} = A_B(y_i^0) \left[m_0^2 T_B(y_i^0) + 2 m_0 T_B^2(y_i^0) + 2 T_B^3(y_i^0) \right] \tag{7.5}$$

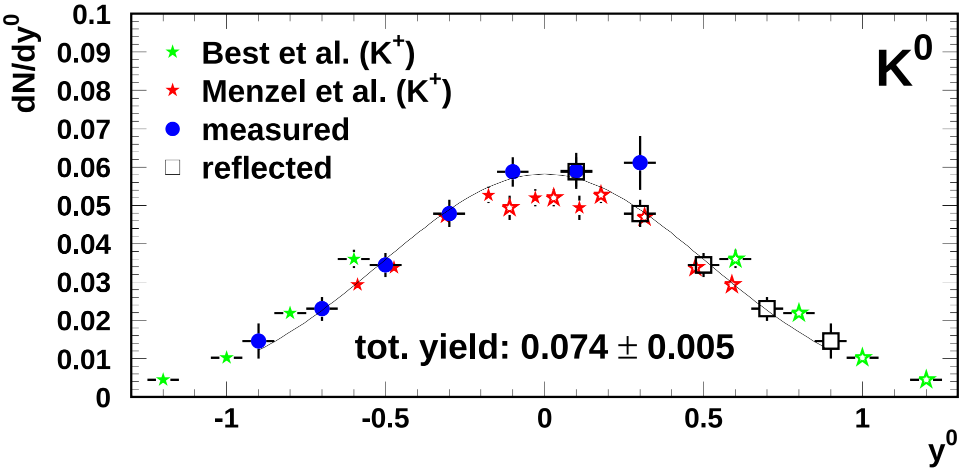


Figure 7.5: Differential yield distribution of K^0_S candidates. The blue circles indicate measured data, the hollow black squares are data points reflected at mid-rapidity. The line represents a Gaussian fit from which the total yield is extracted. The red and green symbols show K^+ distributions taken from [49, 50].

for a specific bin y^0_i (m_0 is the particle's rest mass). Using this method, contributions to the integral from outside the acceptance are included.

The resulting dN/dy^0 distribution for the K^0 is plotted in figure 7.5 which nicely illustrates the complete coverage of the relevant rapidity range. Blue circles and hollow black squares indicate measured and reflected data, respectively. After correction for the reconstruction efficiency the distribution was scaled according to the branching ratio (68.6 %) and the K^0 mixing (50 % K^0_L). The $\overline{K}^0$ has not been taken into account because its yield has the same magnitude than that of the K^-, which is less than 5 % of the observed K^+ yield. Results of previous experiments performed by the KaoS and FOPI collaborations [49, 50] on K^+ production are depicted by the red and green symbols, respectively. The K^0 and the K^+ distributions agree within the given errors.

By fitting a Gaussian distribution (black line) to the data, the total particle yield and the width of the dN/dy^0 distribution are determined. For the K^0 distribution a width of 0.509±0.025 is found. The integration of the Gaussian using the fit parameters results in a total K^0 yield of 0.074±0.005 per event for the applied centrality selection. The combined K^+ distribution has a width of 0.548±0.006 and a total yield of 0.074±0.002.

Figure 7.6 presents the dN/dy^0 distribution of the Λ^0. Again, red cir-

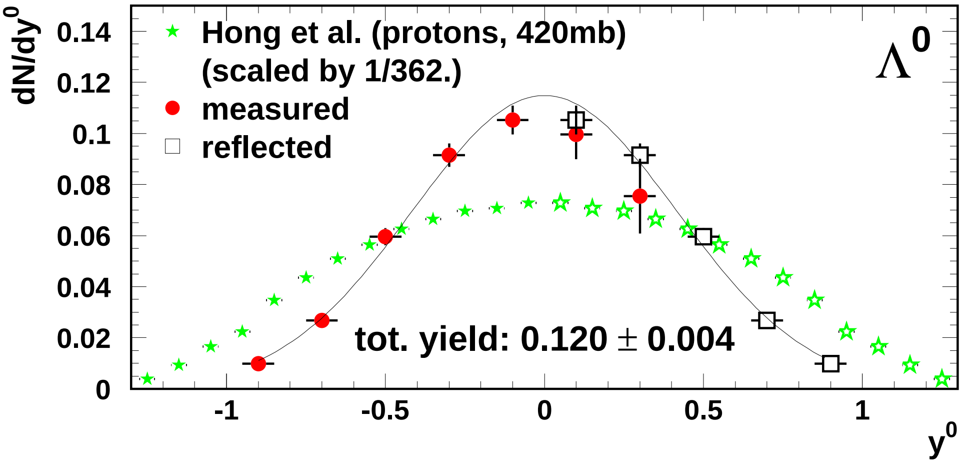


Figure 7.6: Differential yield distribution of Λ^0 candidates. The red circles indicate measured data, the hollow black squares are data points reflected at mid-rapidity. The line represents a Gaussian fit from which the total yield is extracted. The green symbols show the proton distribution from [102].

cles symbolize measured data and hollow black squares denote reflected data points. Similar to the case of the K^0 also the Λ^0 distribution was scaled by its branching ratio (63.9 %). Integrating according to the method described above gives a total yield of 0.120±0.004 and a width of 0.416±0.007. This production probability contains the contributions from the real Λ^0 and from the Σ^0. The latter particle decays into a Λ^0 and a photon with a branching ratio of 100 %. Thus, Λ^0s coming from a decay of a Σ^0 can not be distinguished from primary Λ^0s.

The green stars indicate a proton distribution from [102] which was scaled to have the same integral as the measured data. Unlike the Λ^0s, the protons exhibit a significant longitudinal spread. A hint on this was already seen in the slope parameter distribution. One explanation could be the partial stopping in Ni+Ni reactions at 2 AGeV [41] which leads to an elongated rapidity distribution of the protons. Unlike protons, the Λ^0s are produced in the fireball and scatter with the surrounding nuclear medium resulting in the different shape of their distribution.

7.1.4 Estimation of Systematic Errors

An estimation of the systematic error for the effective temperatures, the total yield and the width of the dN/dy^0 distributions is made by comparing results for different sets of cuts and by explicitely varying certain selection criteria. Table 7.1 presents the results for various sets of cuts which e.g. use different thresholds for d_0, hit multiplicity or p_t. As shown in chapter 5, table 5.4, the resulting signal-to-background ratio differs by a factor 2-3. While the effective temperatures vary only within 1 %, the widths differ by about 4-5 %. The total yield of the K^0 decreases by 7 % when going from the '1' to the '2' cut set. For the Λ^0 a difference of 5 % is found between the cut sets '3' and '1'.

cuts		**T$_{eff}$ [MeV]**	**width**	**tot. yield**
K^0_S	**1**	113 ± 1	0.509 ± 0.025	0.074 ± 0.005
	2	114 ± 1	0.486 ± 0.020	0.069 ± 0.004
Λ^0	**1**	127 ± 1	0.416 ± 0.007	0.120 ± 0.004
	2	127 ± 1	0.435 ± 0.008	0.117 ± 0.003
	3	128 ± 1	0.419 ± 0.010	0.114 ± 0.004

Table 7.1: Comparison of the results for different sets of cuts. The sets used in the final analysis are '1' (K^0) and '1' (Λ^0). A complete description of all sets is given in appendix E.

To a certain extent these differences originate from cuts on quantities which cannot be modeled correspondingly in the simulation. The track quality in the x-y plane, σ_{dxy}, is an example. Lowering the upper threshold for this quantity helps to increase the signal-to-background ratio but also affects the real data stronger than the simulated data. Thus the efficiency correction remains nearly unchanged whereas the raw yield in the real data drops.

Other sources of systematic errors are due to the event sample used for the efficiency correction, the centrality selection and specific cuts, in particular the d_0 cut:

- **Event sample:** The efficiency correction is determined from IQMD events with a certain impact parameter distribution. Going from a correction with $b < 5$ fm to one with $b < 7$ fm, both with a weight on the impact parameter, affects the total yield and T$_{eff}$ at a level of 1 %. While the width of the dN/dy^0 distribution of the Λ^0 changes by 1 %, that of the K^0 decreases by 5 %.

- **Centrality selection:** The centrality selection is done by setting lower thresholds for the Plastic Wall charged particle multiplicity. Changing thresholds for simulated and real data will influence the efficiency correction as well as the raw particle yields. Varying both thresholds by

±1 unit in PMUL causes a 3 % change of the total yield and a 1 % change of the width while T_{eff} is essentially unchanged.

- **Variation of the d_0 cut:** The d_0 cut is the most important cut for the selection of secondary particles. A change of ±0.6 cm in the lower d_0 threshold (which is substantial) results in a 3 % change of the total yield, while the effective temperature is not altered (<1 %). The width of the dN/dy^0 distribution varies by 2 %.

To obtain the total systematic error all contributions are added quadratically. The results are presented in table 7.2.

	K^0 systematic error		**Λ^0 systematic error**	
quantity	**relative**	**absolute**	**relative**	**absolute**
T_{eff} [MeV]	1.7 %	2	1.7 %	2
width	7.1 %	0.036	5.1 %	0.021
tot. yield	8.2 %	0.006	6.6 %	0.008

Table 7.2: Estimated systematic errors of the effective temperature T_{eff}, the total yield and the width of the dN/dy^0 distribution.

7.2 Collective Flow

The collective behaviour of the produced strange particles is studied by analyzing their azimuthal distributions with respect to the reaction plane. Two kinds of collective flow are distinguished in near-central collisions. One is the in-plane emission of particles, the *directed flow*, which is due to the deflection of the spectators by scattering with the participants. The other one is the out-of-plane emission, the *squeeze-out*, which originates from particles from the fireball which are deflected into different regions of phase space by spectator matter or by pressure gradients within the fireball. Thus, depending on the centrality, asymmetries can be observed in the emission of particles with respect to the reaction plane. The normalized azimuthal distribution $dN/d\phi$ is expressed as follows:

$$\frac{dN}{d\phi} = \frac{1}{2\pi}\left(1 + \sum_{n=1}^{\infty} 2v_n \cos\left(n\phi\right)\right) \quad , \tag{7.6}$$

where $v_n = \langle\cos(n\phi)\rangle$ is the n-th Fourier coefficient. Non-zero coefficients indicate deviations from an isotropical emission of particles. The first and the second coefficients, v_1 and v_2, are commonly used to study the particle emission patterns. While v_1 is more sensitive to the mean field and scattering effects, v_2 yields information about pressure gradients and the reaction dynamics.

Collective effects in the reaction plane are called *side flow* or *directed flow* and are described by v_1. This quantity can be calculated from the transverse momentum p_t and the azimuthal emission angle ϕ_{em} (w.r.t. the reaction plane):

$$v_1 = \langle \cos(\phi_{em}) \rangle = \left\langle \frac{p_x}{p_t} \right\rangle \tag{7.7}$$

with the in-plane transverse momentum component p_x. In order to compare different particle species it is more convenient to use $\langle p_x \rangle / m_0$ instead of v_1.

The *elliptic flow* or *squeeze-out* refers to azimuthal asymmetries perpendicular to the reaction plane. In central collisions, the azimuthal emission is nearly symmetric and no elliptic flow is observed. The corresponding coefficient v_2 is calculated from the transverse momentum components p_x and p_y via

$$v_2 = \langle \cos(2\phi_{em}) \rangle = \left\langle \cos^2(\phi_{em}) - \sin^2(\phi_{em}) \right\rangle = \left\langle \frac{p_x^2 - p_y^2}{p_t^2} \right\rangle \ . \tag{7.8}$$

All spectra of flow observables (v_1, v_2) in the following subsections are background subtracted according to the method explained in chapter 5, section 5.4 and have been corrected according to the method developed by Ollitrault (cf. chapter 4 and [94]). In addition a cut of $p_t/m_0 > 0.5$ has been applied to all y^0-dependent spectra in order to avoid distortions by the geometrical acceptance of the CDC. Due to this cut, particles with transverse masses smaller than about 0.12 GeV/c^2 are discarded. The reconstruction efficiency at higher m_t is approximately constant. Therefore, no efficiency correction was performed.

7.2.1 Directed Flow

To determine the directed flow of a particle, spectra of the quantity p_x/p_t are filled and the mean value

$$v_1 = \frac{1}{N} \sum_{i=1}^{N_{bin}} n_i \cdot \left[\frac{p_x}{p_t} \right]_i \ , \qquad N = \sum_{i=1}^{N_{bin}} n_i \tag{7.9}$$

is extracted, where N_{bin} denotes the number of bins in the spectrum, n_i the number of counts in the i-th bin and $[p_x/p_t]_i$ the position of the i-th bin. The

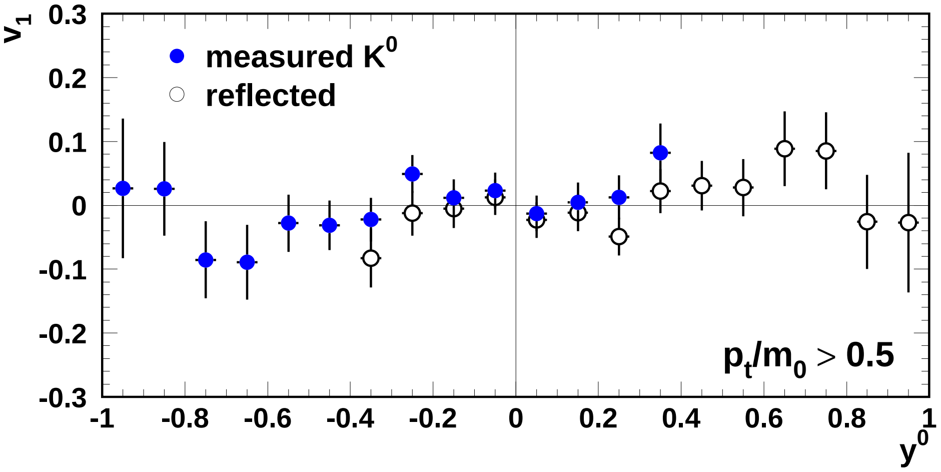


Figure 7.7: Directed flow v_1 of the K^0 as function of y^0. Blue circles depict the measured data, hollow black squares the reflected data. To avoid acceptance effects, an additional cut of $p_t/m_0 > 0.5$ was applied.

error σ_{v_1} of the mean value can then be calculated by using

$$\sigma_{v_1} = \sqrt{\sum_{j=1}^{N_{bin}} \left(\frac{\partial v_1}{\partial n_j} \cdot \sigma_{n_j} \right)^2}, \qquad \frac{\partial v_1}{\partial n_j} = \frac{1}{N} \left(\left[\frac{p_x}{p_t} \right]_j - v_1 \right) \tag{7.10}$$

with the errors σ_{n_j} of the different bin contents. The same type of formula is also used for the calculation of the $\langle p_x \rangle$ and v_2 errors.

Figure 7.7 presents the background-corrected spectrum of the directed flow as function of y^0 for the K^0. As before, the blue symbols indicate the measured data and the hollow black symbols indicate the data reflected at mid-rapidity. Within the errors, no dependence of v_1 on the rapidity is visible which is in agreement with earlier analyses concerning K^+ and K^0 [104].

Due to tracking inefficiencies the K^0 data were shifted along the v_1 axis. This shift was also observed when analyzing reconstructed K^0 in the simulated events. The observations are described in subsection 7.2.2. A similar observation was also made by the EOS collaboration [105]. The magnitude of the shift is higher for the K^0 than for the Λ^0 (see below) while for the protons no shift is observed. Therefore, the shift seems to be caused by pions. A shift of 0.049 was determined by fitting a constant to the spectrum before Ollitrault correction. Fitting a linear function yielded a slope of 0.022 and an intercept on 0.051. Therefore, with the given errors, the assumption of a constant v_1 for the K^0 is justified and the shift is applied.

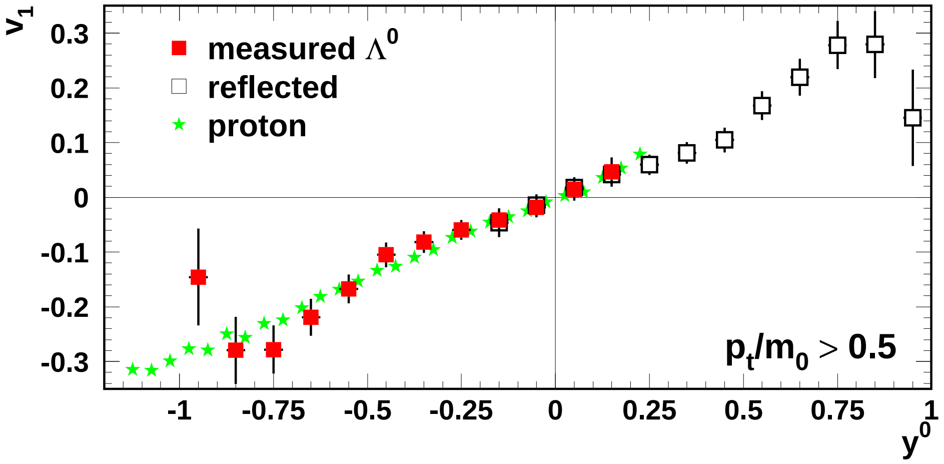


Figure 7.8: Directed flow v_1 of the Λ^0 as function of y^0. Red circles depict the measured data, hollow black squares the reflected data. The signal of proton coming from the same data sample is indicated by the green stars. To avoid acceptance effects, an additional cut of $p_t/m_0 > 0.5$ was applied.

The Λ^0 exhibits a totally different flow behaviour which is depicted in figure 7.8. In the backward hemisphere v_1 is negative, rising towards positive values in the forward hemisphere. Similarly to the K^0 a shift of -0.018 on v_1, determined by fitting a linear function around mid-rapidity, was used to align the distribution to zero directed flow at mid-rapidity.

The Λ^0 flow behaviour is also seen for protons depicted by the green stars in the picture. While the directed flow of both particle species is found to agree in this work, other observations have been made by the EOS and E895 collaborations [105, 106] finding a weaker Λ^0 flow in comparison to the protons.

From the $\langle p_x \rangle / m_0$ distributions of Λ^0 and proton which are not shown here, the flow parameter F (slope of that distribution at mid-rapidity) can be determined. For the Λ^0, a value of $F_\Lambda = 0.141 \pm 0.035$ is found within the interval -0.2 $< y^0 <$ +0.2 while the proton flow parameter is $F_p = 0.155 \pm 0.015$ for the same interval. Thus, within the errors, both parameters agree and can be compared to the findings of other experiments later on.

Using the transverse momentum dependence of v_1, the *differential flow* can be examined in a rapidity region where a pronounced directed flow signal is observed like in the case of the Λ^0 or the protons. In a symmetric system this is true either at target or projectile rapidities. In addition, although

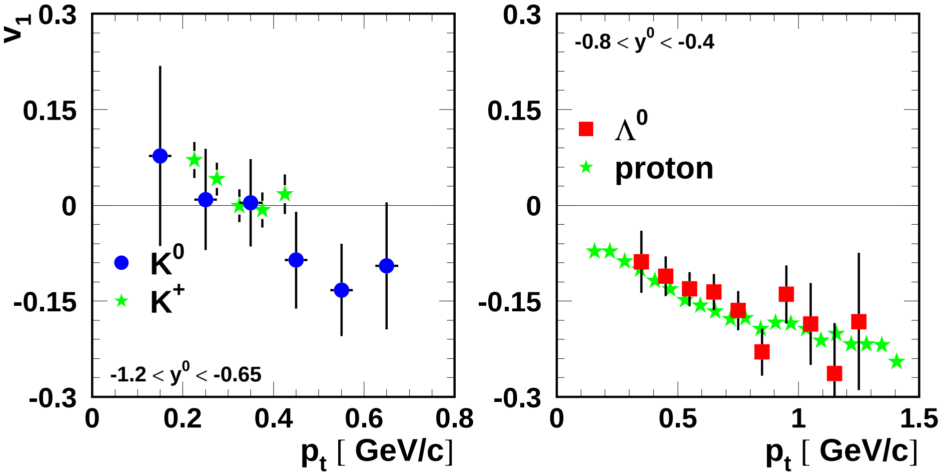


Figure 7.9: Differential flow $v_1(p_t)$ of K^0 (left) and Λ^0 (right) in the interval -1.2 < y^0 < -0.65. The green stars in the K^0 plot depict results from an earlier measurement of K^+ [54] and the stars in the Λ^0 plot indicate the behaviour of protons under the same conditions.

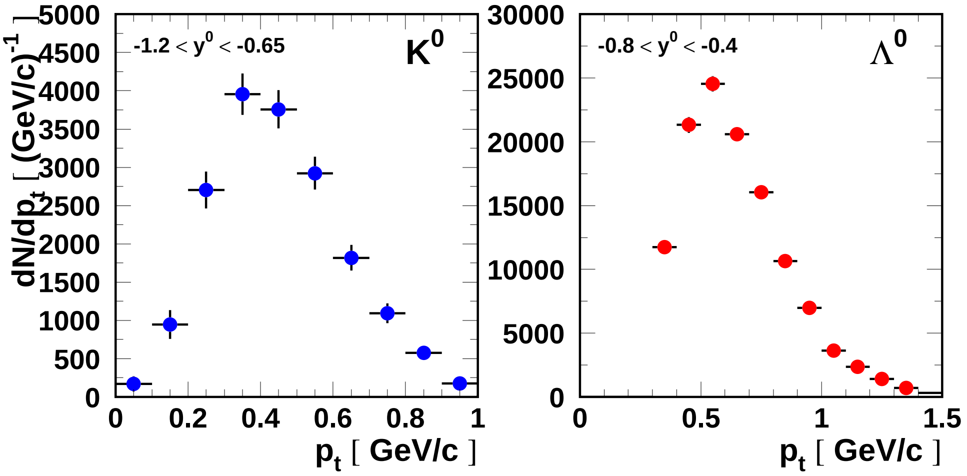


Figure 7.10: Transverse momentum spectra of K^0 and Λ^0 candidates for the differential flow analysis within the given rapidity ranges.

there is no directed flow signal, the p_t dependence of v_1 for the K^0 is of interest. Results of a K^+ analysis [54] find a change from antiflow to flow for increasing transverse momenta.

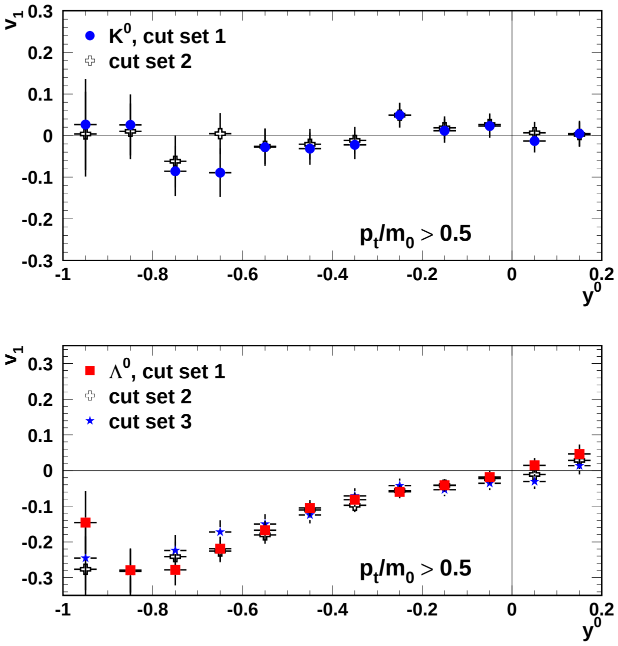


Figure 7.11: Comparison of the directed flow signal of K^0 (upper panel) and Λ^0 (lower panel) for different sets of cuts. Within the errors, the results agree nicely thus ruling out a strong influence of the combinatorial background and the background subtraction procedure.

The differential flow of K^0 in the rapidity range from -1.2 to -0.65 is presented in the left plot of figure 7.9. The same p_t dependence as for the previously measured K^+ is found, which is only expected in case of a repulsive K^+N in-medium potential. Due to the different target position in the 2003 experiment, the acceptance window of the CDC was shifted towards mid-rapidity. Therefore, less than 10 % of all reconstructed K^0 candidates come from the rapidity range given above (cf. figure 5.9 on page 65).

In order to get more conclusive results for the Λ^0, the rapidity window was moved to the range -0.8 $< y^0 <$ -0.4, where sufficient statistics are available and where v_1 still is significantly different from zero. The results are presented in the right plot of figure 7.9. As for the rapidity-dependent v_1 distribution, protons and Λ^0 show the same behaviour. This suggests that the directed flow of those particles agrees for each cell (p_t, y^0) of the phase space. Thus, the question of differences between $U_{\Lambda N}$ and U_{NN} needs further investigation by detailed comparisons to model calculations.

The background-corrected p_t spectra of K^0 and Λ^0 candidates in the rapidity ranges mentioned above are plotted in figure 7.10. Comparing the spectra for the different ranges one finds that, despite the similar reconstruction efficiencies, the available statistics increases by a factor of six when going from -1.2 $< y^0 <$ -0.65 to -0.8 $< y^0 <$ -0.4. To get conclusive results for the differential flow of the K^0 at target rapidity, a high-statistics experiment with the target in its nominal position has to be done.

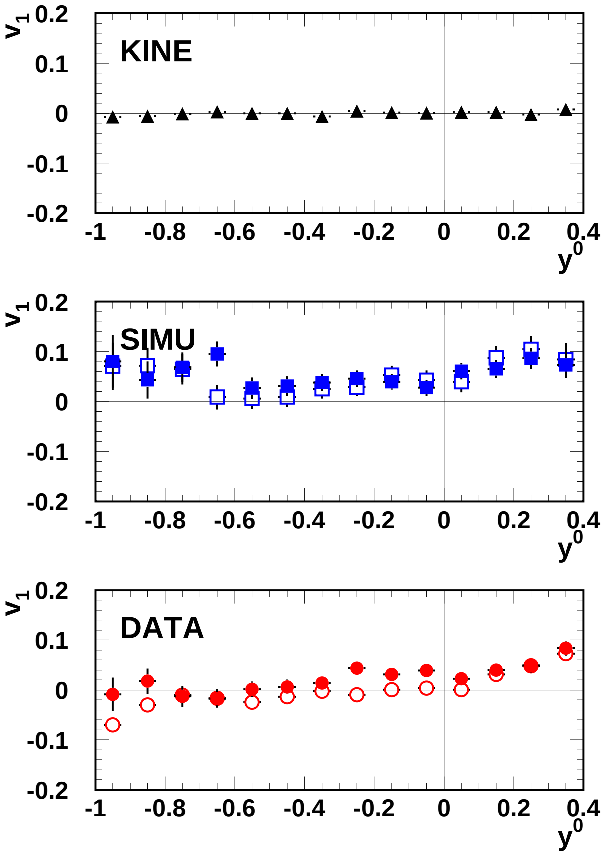


Figure 7.12: Illustration of the v_1 shift for the K^0. The upper, middle and lower plots show the directed flow of the thermally generated particles (KINE), of the reconstructed particles from the simulated events (SIMU) and of the reconstructed particles from the measured data (DATA), respectively. The open symbols indicate the flow of the mixed-events background. The higher values of v_1 found at rapidities larger than 0.2 are due to the geometrical acceptance of the CDC.

7.2.2 Systematic Effects

Like the kinematic quantities, the flow observables can be subject to systematic errors. By studying their behaviour under different sets of cuts with different signal-to-background ratios, an estimate of systematic deviations and the influence of the combinatorial background is possible.

Figure 7.11 shows the directed flow of K^0 (upper panel) and Λ^0 (lower panel) for the different cut sets. Despite the different signal-to-background ratios of the cut sets the v_1 distributions exhibit a nice agreement for the respective particles. The same observation is made when looking at the p_t-dependence of the directed flow or the v_2 distributions. Therefore, an influence of the combinatorial background on the signals can be excluded and the application of the background correction introduces no artificial flow signal.

An illustration of the v_1 shift mentioned above is plotted in figure 7.12 for the K^0. No background and Ollitrault correction have been performed for the data presented in these plots. The cut on $p_t/m_0 > 0.5$ was applied. The thermally generated particles (KINE) are emitted isotropically and therefore do not show any flow behaviour. The shift towards positive values of v_1 which was found for the measured data (DATA) is also present (and even more pronounced) for the K^0 reconstructed from simulated events (SIMU). The fact that the v_1 shift is found for simulated and measured events suggests that it is a consequence of tracking inefficiencies. Within the errors, K^0 reconstructed from simulated events show a constant v_1 as a function of y^0 which proves that apart from the shift, no flow signal is generated by the reconstruction method. Therefore, the effect is corrected by shifting back the measured v_1 distribution by the offset determined from a constant or linear fit.

The same behaviour is found for the Λ^0 although the magnitude of the shift is significantly smaller. This means that the shift probably originates from tracking inefficiencies concerning pions.

Chapter 8

Discussion and Interpretation

Strange particles are an essential probe to study in-medium effects in nuclear matter. In order to find modifications of hadronic properties, a detailed confrontation with theoretical model calculations is necessary. This chapter will first provide a survey of already existing results concerning (neutral) strange particle production at conditions similar to those at SIS/GSI. Afterwards, the experimental results are compared to expectations derived from transport codes. For the interpretation of measured particle ratios, predictions of a statistical model are also considered and the balance of produced strange particles in the Ni+Ni system at 1.93 AGeV beam energy will be discussed.

8.1 Comparison to existing Data

The results presented in the previous chapter can now be compared to those published by other experiments. Data from previous KaoS and FOPI experiments were already used for consistency checks. Thus, a short summary will be given here.

First hints for in-medium effects were observed in the subthreshold production of charged kaons [48, 47]. Indications for a slightly repulsive K^+-N potential and an attractive K^--N potential have been found in the K^+ differential flow and the kinetic energy dependence of the K^-/K^+ ratio [54, 55]. However, regarding the ability of the models to reproduce the measured K^+ differential yield distribution [49, 50], it becomes clear that no decisive answer can be given without extensive knowledge of the cross sections used in the code. The dN/dy^0 distribution of the K^0 presented in this work confirms the K^+ results and provides another starting-point for model comparisons.

The situation for the K^- is similar. Recent analyses suggest that as much as 20 % of the observed K^- come from decays of ϕ mesons [69] and

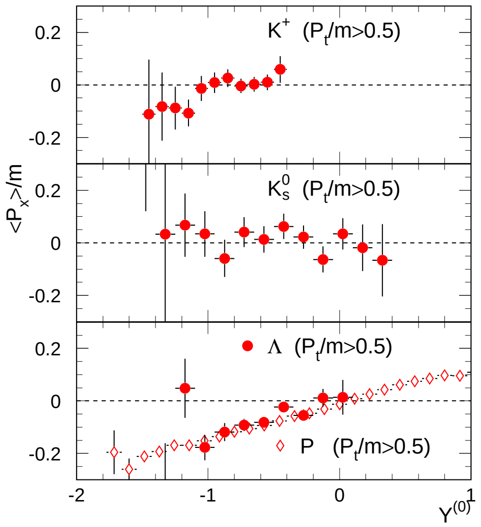


Figure 8.1: Directed flow of K^+, K^0 and Λ^0 (with protons) as published in [104]. Plotted is $\langle p_x \rangle / m_0$ as a function of y^0 under a cut on $p_t/m_0 > 0.5$. Within the errors, both kaons show the same flow pattern. Λ^0 and proton flow also agree.

that strangeness exchange via $\pi\Lambda^0 \rightarrow K^-$N may play an important role for the K^- production [56]. Therefore, the Λ^0 results presented in this work again in combination with model calculations can give some clues on the exchange mechanism.

An early investigation of K^0 and Λ^0 directed flow with the FOPI detector [104] showed that Λ^0 and proton flow have approximately the same magnitude (cf. figure 8.1). The K^+ and K^0 flow were found to agree in being compatible with zero. Due to the low statistics of the neutral strange particles an extraction of production yields was not possible at that time. The flow results of this work confirm the previous findings with a better accuracy.

At the AGS, the production of strange particles was studied at beam energies from 2 to 8 AGeV. First results concerning the Λ^0 were presented by the EOS collaboration [105] for the Ni+Cu system at an energy of 1.97 AGeV. For transverse masses $m_t - m_0$ <0.5 an effective temperature of 106 ± 5 MeV was reported for the Λ^0 and 142 ± 1 MeV for the protons in an interval of -0.25 $< y^0 <$ +0.25 around mid-rapidity. In this work, a value of 127 ± 1 MeV was determined, which agrees with the proton temperature of 125 MeV found under the same conditions [102]. The disagreement in the Λ^0 temperatures could be a result of the lower statistics in the EOS experiment which limits the m_t spectrum to masses smaller than 0.5 GeV/c^2.

In the same publication by the EOS group, the slope of the directed flow of the Λ^0 at mid-rapidity was found to be somewhat smaller than that of the pro-

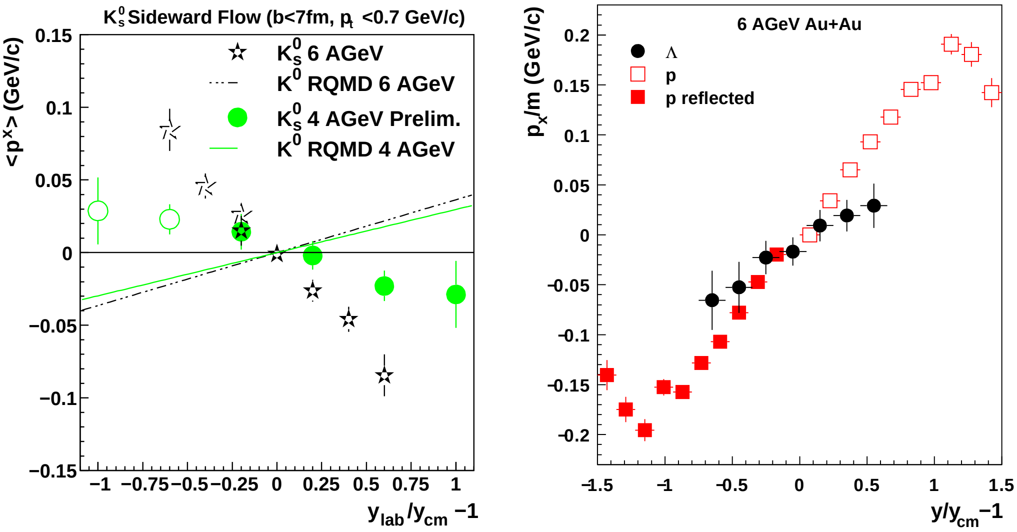


Figure 8.2: Directed flow of K^0 and Λ^0 measured by the E895 collaboration [107, 106]. The K^0 exhibit an antiflow behaviour while the Λ^0 shows a flow signal which is smaller than that of the protons.

ton. The quantity $(d\langle p_x\rangle/dy^0)\cdot(m_p/m_0)$ (m_p: proton mass, m_0: mass of the considered particle) is 133 ± 10 MeV for the protons and about 85 ± 43 MeV for the Λ^0. Multiplying the slopes of the $(d\langle p_x\rangle/m_0)$ vs. y^0 spectra from this analysis with the proton mass[1], values of 145 ± 14 MeV and 132 ± 33 MeV are found for protons and Λ^0, respectively. Within the errors these findings agree and reproduce the slope of the protons reported by EOS. Their result for the lambda cannot be confirmed.

Later AGS experiments performed by the E895 collaboration did systematic studies of neutral strange particle production and flow in Au+Au at beam energies from 2 to 8 AGeV [106]. Using the number of reconstructed particles per event and the reconstruction efficiencies given in the publication, a rough estimation of the total particle yields can be done. For a beam energy of 2 AGeV the total Λ^0 production probability is about 0.2 per event and that of the K_S^0 is approximately 0.11 per event, respectively. Given the substantially larger system (Au instead of Ni) and the less central event selection these numbers are hard to compare. However, the magnitude of the results agrees between both experiments.

Analyses of the directed flow of Λ^0 and K^0 were done for impact parameters smaller than 7 fm [106, 108, 107]. The Λ^0 flow depicted in the left plot of figure 8.2 shows the same qualitative behaviour as in the EOS results. With increasing beam energy, the Λ^0/p flow ratio decreases which is

---

[1] This is done in order to compare to the EOS results

attributed to the influence of the Λ^0-N potential. At 2 AGeV, the flow ratio is approximately 0.6 for the Au+Au system compared to that of 0.75 for Ni+Cu measured by EOS. Within the errors both experiments agree. Taking the slopes of proton and Λ^0 reported earlier in this work, a flow ratio of 1.1 ± 0.3 is determined which, within the errors agrees with the EOS result but can also not exclude ratio published by the E895 collaboration for the beam energy of 2 AGeV.

A pronounced antiflow behaviour was reported for the K^0 at 6 AGeV. Model calculations only regarding scattering processes fail to describe this observation which supports the idea of a repulsive K-N potential. The magnitude of the antiflow effect decreases with decreasing beam energy. Furthermore, the same qualitative behaviour is found for the K^+ while the directed flow of the K^- is compatible with zero [107]. The analysis in this work finds no directed flow of the K^0. Despite the smaller system and the different centrality selection, this finding would suggest an evolution of antiflow with increasing beam energy.

8.2 Model Comparisons

In this section the results presented in the previous chapter are discussed with respect to predictions of various models. Kinematic quantities are compared to transport model predictions from UrQMD (V1.3) [109] and from IQMD [51]. While UrQMD uses a hard EOS without momentum-dependent interactions, a soft EOS and momentum-dependent interactions are included into IQMD. Two different versions of IQMD are used. One with in-medium potentials (labeled 'w. pot') for the K^+, and another version without in-medium potentials (labeled 'no pot.'). Due to low statistics in case of UrQMD (which are a consequence of the non-perturbative approach), the observed directed flow is compared to IQMD calculations only. Finally, measured particle ratios are discussed taking into account predictions from the transport models mentioned above and from a statistical model [5] using $T = 63$ MeV and $\mu_B = 784$ MeV. Note that mentioning K^0 implies that K^0 and $\overline{K}^0$ have been taken into account (due to the mixing). In the case of the Λ^0, the Σ^0 is also included. In the experiment, Λ^0 and Σ^0 are anyway measured inclusively and the K^0_L contribution to the K^0 is included in the efficiency correction.

8.2.1 Kinematic Quantities

The effective temperature, the width of the dN/dy^0 distribution and the total yield of K^0 and Λ^0 are compared to QMD model calculations. While the

K^0	**model / data**	**impact parameter**		
		$b <$ **3.3 fm**	$b <$ **4.0 fm**	$b <$ **4.5 fm**
T$_{eff}$	UrQMD (v 1.3)	93	93	94
	IQMD (w. pot.)	111	110	109
	IQMD (no pot.)	100	99	98
	data	**112 ± 1 ± 2**		
width	UrQMD (v 1.3)	0.533	0.545	0.530
	IQMD (w. pot.)	0.617	0.618	0.618
	IQMD (no pot.)	0.580	0.587	0.591
	data	**0.510 ± 0.025 ± 0.036**		
yield	UrQMD (v 1.3)	0.115	0.101	0.091
	IQMD (w. pot.)	0.126	0.112	0.102
	IQMD (no pot.)	0.196	0.173	0.157
	data	**0.074 ± 0.005 ± 0.006**		

Table 8.1: Model predictions for effective temperature, width of the dN/dy^0 distribution and total yield of the K^0. The calculations were done for different impact parameters. The experimental results from this work are included with their statistical and systematical errors.

experimental results are given for 350 mb of the total reaction cross section, the model calculations are evaluated for three different impact parameter intervals in order to see the dependence of the observables on the centrality selection. Using a sharp cutoff model, the choice of $b < 3.3$ fm, $b < 4.0$ fm and $b < 4.5$ fm corresponds to cross sections of 340 mb, 500 mb and 640 mb, respectively.

Model calculations and experimental results for the K^0 are presented in table 8.1. The effective temperatures at mid-rapidity predicted by IQMD and UrQMD do not show a strong dependence on the impact parameter selection. Among the different models, variations of about ±10 % are found for T$_{eff}$. Within the experimental centrality selection, only the IQMD model using in-medium potentials is able to reproduce the result of 112 MeV. Model calculations of the dN/dy^0 widths exhibit fluctuations of less than 2 % when varying the impact parameter. Differences between the models again are on the level of ±10 % with the largest width predicted by the IQMD model with in-medium potentials. The width observed in the experiment is lower than any of the model calculations. Within the errors the data only agree with UrQMD predictions. The predicted yields drop by 20 % when changing the impact parameter range from less than 3.3 fm to less than 4.5 fm for all models. The highest values are found for IQMD without in medium potentials, where due to the absence of repulsion, more K^0 are produced. The experimental finding is much lower than any model prediction.

A compilation of model expectations and experimental results for the Λ^0 is given in table 8.2. Variations between different model calculations and centrality dependence of the effective temperature are comparable to those of the K^0. However, the experimental result is now higher than any of the predictions. For the widths of the yield distributions, the situation is different in the case of the Λ^0. Both versions of IQMD predict the same width which is due to the fact that no in-medium potential was used for this particle. Furthermore, no centrality dependence of the width can be found. UrQMD predicts a width which is significantly larger and which seems to increase when including more peripheral events. Owing to the low statistics of those model calculations this effect is on the level of the fit errors. The measured value roughly falls in the middle between UrQMD and IQMD predictions. Estimates of the total yield exhibit the same qualitative behaviour like those for the K^0. Again, the IQMD calculation without potentials has the highest value which can be understood because $K^{0,+}$ and Λ^0 (Σ^0) are linked by their associated production. The measurement lies below all predictions and, regarding the errors, only agrees with the UrQMD calculation.

Λ^0		**impact parameter**		
	model / data	$b <$ **3.3 fm**	$b <$ **4.0 fm**	$b <$ **4.5 fm**
T$_{eff}$	UrQMD (v 1.3)	109	110	108
	IQMD (w. pot.)	94	93	93
	IQMD (no pot.)	98	97	97
	data	**127 ± 1 ± 2**		
width	UrQMD (v 1.3)	0.525	0.531	0.555
	IQMD (w. pot.)	0.337	0.338	0.339
	IQMD (no pot.)	0.343	0.344	0.344
	data	**0.461 ± 0.007 ± 0.021**		
yield	UrQMD (v 1.3)	0.131	0.118	0.107
	IQMD (w. pot.)	0.166	0.147	0.134
	IQMD (no pot.)	0.274	0.243	0.220
	data	**0.120 ± 0.004 ± 0.008**		

Table 8.2: Model predictions for effective temperature, width of the dN/dy^0 distribution and total yield of the Λ^0. The calculations were done for different impact parameters. The experimental results from this work are included with their statistical and systematical errors.

Figure 8.3 presents model predictions and experimental results for the

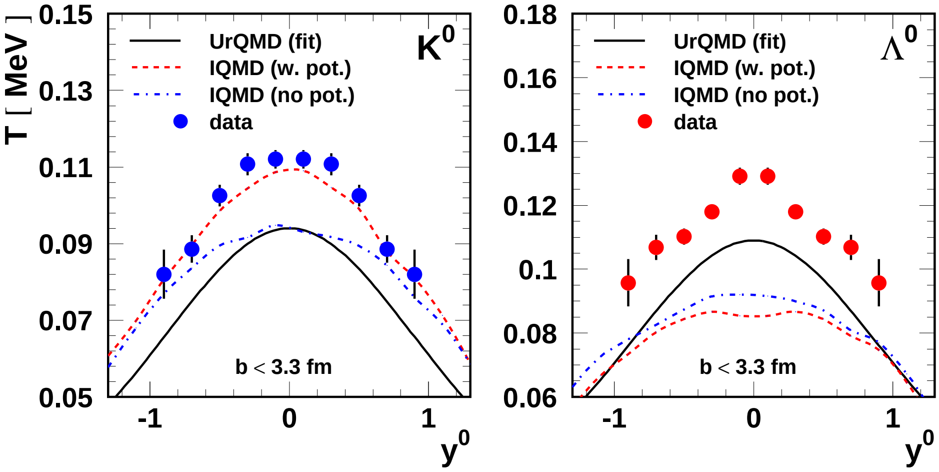


Figure 8.3: Comparison of the inverse slope parameters between measured data (K^0, Λ^0) and different model calculations.

slope parameter distributions for both, K^0 (left) and Λ^0 (right). Due to the low statistics and the resulting fluctuations of T_B in the case of UrQMD, a fit with equation 7.2 is plotted. The IQMD model using an in-medium potential for the kaons reproduces the measured slope parameter distribution of the K^0, while the version without potentials and the fit to UrQMD results in a value which is significantly lower. Comparing both IQMD parameter sets, one finds that the presence of in-medium effects mainly influences the slopes at mid-rapidity while the calculations agree when going beyond target or projectile rapidities. None of the models succeeds in describing the corresponding distribution for the Λ^0.

Experimental results and model calculations for the differential yield distributions are shown in figure 8.4. The K^0 lies below all predictions. This is interesting since an IQMD calculation with in-medium potentials reproduced the previously measured K^+ distribution [53] which agrees nicely with the K^0 determined in the present work. Here, the UrQMD distribution agrees with IQMD using the in-medium K^+ potential while IQMD without potential predicts a value which, due to the absence of the repulsive potential for the kaons, is far too large. Regarding the Λ^0 dN/dy^0 distribution, both IQMD version fail in describing the experimental results. Their expected yields are too high and the widths of the distributions are too narrow. UrQMD describes the results reasonably well.

Regarding all comparisons made above one can summarize that, at present,

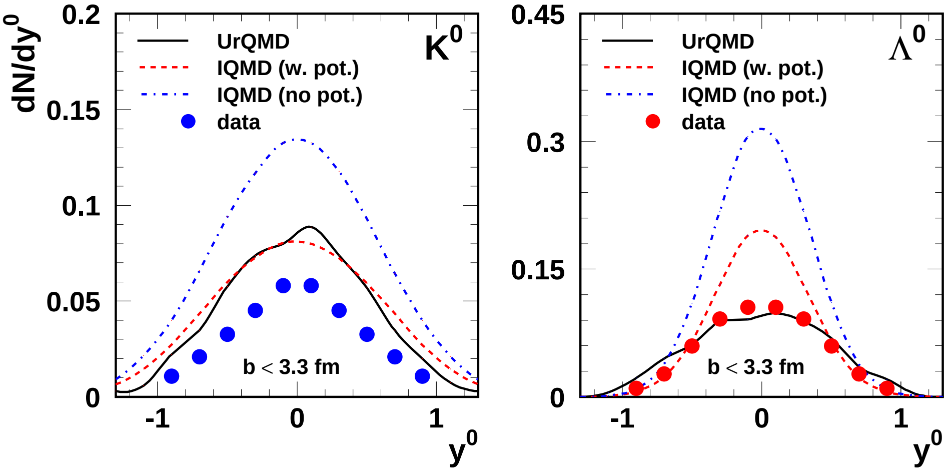


Figure 8.4: Comparison of the differential yield distributions between measured data (K^0, Λ^0) and different model calculations.

none of the models is able to provide a consistent description of the experimental observations.

8.2.2 Directed Flow

The measured directed flow of K^0 and Λ^0 is compared to IQMD calculations in figure 8.5. Both versions of the IQMD model also predict a very weak directed flow in accordance with the experimental findings. Using an in-medium potential for the K^+ a slight anti-flow of the K^0 is expected while without potentials a minor flow signal is observed. Considering the experimental errors and the small effects predicted by the models, no conclusions can be drawn.

The measured flow of the Λ^0 agrees with both model calculations which shows that, despite the associated production with a K^0 or a K^+, the Λ^0 flow is almost insensitive to the presence of an in-medium potential for the kaons. In order to further investigate the directed flow of the Λ^0, model calculations with in-medium potentials for the Λ^0 itself are needed. An obvious prerequisite is that they should properly describe the measured yield and slope parameters.

In figure 8.6 the directed and differential directed flow of protons and Λ^0 are compared to IQMD model predictions using the K^+ in-medium potential. The proton directed flow in IQMD is slightly above the measurement, par-

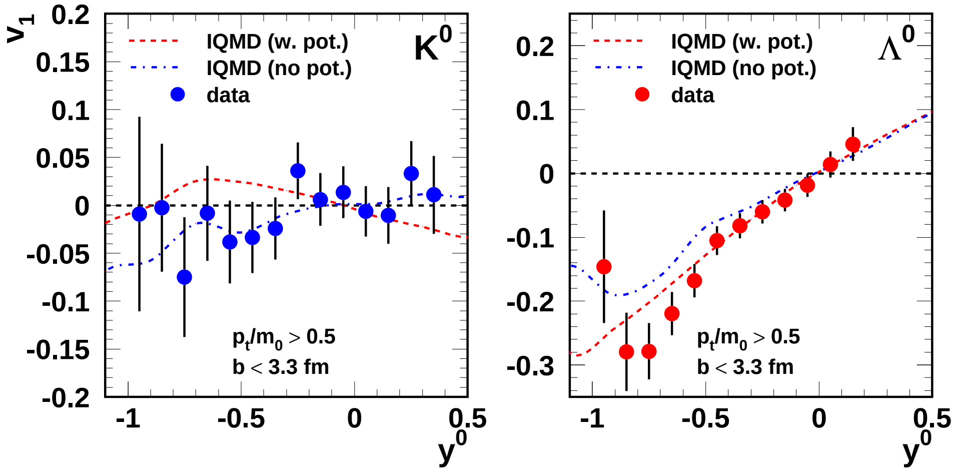


Figure 8.5: Comparison of the measured directed flow of K^0 and Λ^0 with predictions from the IQMD model with and without in-medium potentials for the kaons.

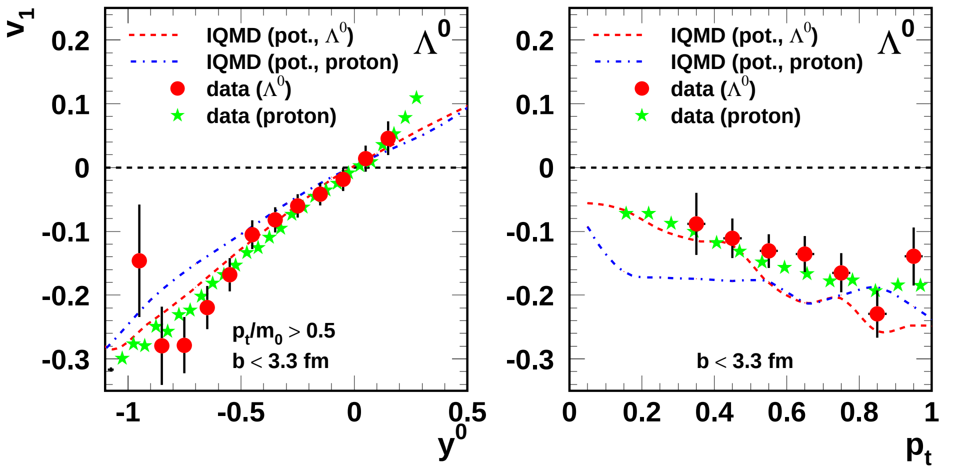


Figure 8.6: Comparison of measured directed flow and differential directed flow for Λ^0 and protons to IQMD calculations.

ticularly at target rapidities. Regarding the differential flow, data and model calculations agree for transverse momenta smaller than 0.5 GeV/c while the expectation for the protons clearly deviates from the measurement. Again one has to stress that updated codes are needed which are able to reproduce

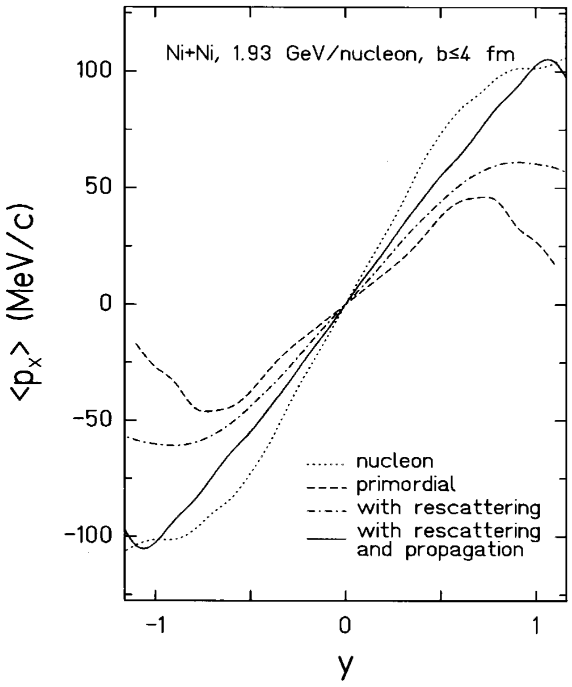


Figure 8.7: Directed flow of the Λ predicted by the RVUU relativistic transport model [110]. The dotted curve denotes the nucleon flow while the dashed, dot-dashed and solid curves indicate the primordial Λ flow, the flow including rescattering and the flow including rescattering and propagation in a Λ-N potential, respectively.

the Λ^0 properties.

The influence of an in-medium Λ-N potential on the directed flow of Λ baryons at SIS energies has already been investigated theoretically for the Ni+Ni system in [110]. Different contributions to the directed flow of the Λ were studied using the RVUU transport model. The primordial flow of the produced Λ is found to be significantly weaker than the nucleon (proton) flow. In order to find a Λ flow signal which agrees with that of the protons it is necessary to include elastic rescattering and the propagation in a mean field potential for the Λ. For the rescattering, a parameterization of the elastic ΛN cross section is applied neglecting inelastic processes. The mean field potential is defined as

$$U_\Lambda(\vec{p},\rho) = \sqrt{(m_\Lambda - \Sigma_S^\Lambda)^2 + \vec{p}^{\,2}} + \Sigma_V^\Lambda - \sqrt{m_\Lambda^2 + \vec{p}^{\,2}} \tag{8.1}$$

using $\Sigma_S^\Lambda \approx 2/3\Sigma_S^N$ and $\Sigma_V^\Lambda \approx 2/3\alpha\Sigma_V^N$ where Σ_S^N and Σ_V^N are the nucleon scalar and vector potential, respectively. The factor 2/3 because according to the constituent quark model the Λ mean field potential is approximately 2/3 of that of a nucleon. The parameter α is introduced to study the sensitivity of the Λ flow to the potential in dense matter. It is varied such that at normal nuclear matter density the magnitude of the mean-field potential agrees with the experimentally determined value for the potential.

While the magnitude of the Λ flow is found to be robust against changes of the ΛN scattering cross section, it shows a dependence on the choice of α. With a decreasing value of α the strength of the vector potential is reduced and, according to the model calculations, the Λ yield and the strength of the flow will increase. Regarding the dependence of the total yield and the flow on the choice of the parameter α in the model, a value of $\alpha \approx 0.7$ is necessary to reproduce the experimental results (total yield: 0.120, $d\langle p_x \rangle / dy^0 = 157$ MeV at $y^0 = 0$). In the framework of the the model this would correspond to a strong mean field potential for the Λ.

8.2.3 Particle Ratios

Particle yield ratios carry information about the degree of chemical equilibration reached in heavy-ion collisions. Table 8.3 presents a comparison between measurements and predictions from QMD codes and a statistical model.

		impact parameter		
ratio	**model / data**	$b <$ **3.3 fm**	$b <$ **4.0 fm**	$b <$ **4.5 fm**
$\frac{K^0}{K^+}$	UrQMD (v 1.3)	1.092	1.067	1.058
	IQMD (w. pot.)	1.079	1.079	1.077
	IQMD (no pot.)	1.051	1.051	1.051
	thermal model	1.074		
	data	**1.000 ± 0.060**		
$\frac{K^0}{\Lambda^0 + \Sigma^0}$	UrQMD (v 1.3)	0.893	0.862	0.820
	IQMD (w. pot.)	0.738	0.736	0.681
	IQMD (no pot.)	0.707	0.706	0.671
	thermal model	0.510		
	data	**0.617 ± 0.039**		
$\frac{\Sigma^+ + \Sigma^-}{\Lambda^0 + \Sigma^0}$	UrQMD (v 1.3)	0.573	0.543	0.559
	IQMD (w. pot.)	0.320	0.323	0.322
	IQMD (no pot.)	0.338	0.337	0.336
	thermal model	0.371		
	data	**0.183 ± 0.053**		

Table 8.3: Comparison of strange particle yield ratios between data and model predictions.

The K^0/K^+ ratio is expected to be balanced: the mass difference of the particles is less than 1 % and in Ni+Ni collisions the N/Z ratio is only slightly less than one. All models predict a ratio which is slightly larger than one. The experimental result is 1.000 ± 0.060 and essentially agrees with all models.

For the $K^0/(\Lambda^0 + \Sigma^0)$ ratio the situation is different. Assuming equal masses K^0 and K^+, equal masses for Λ^0 and all Σs and an isospin-symmetric system, the ratio should be 0.75 when regarding the production dominated by the $NN \rightarrow K^0\Lambda^0(\Sigma^0)N$ channel. Taking into account the mass difference between Λ^0 and all Σ ($\approx$ 80 MeV), the ratio should even increase. The transport model predictions come close to this estimation, although the values given by UrQMD seems somewhat high. From the statistical model, a value is expected which is significantly lower. The measured ratio is about 0.617 ± 0.039, lying in-between the IQMD prediction without in-medium potentials for the kaons and the statistical model result.

The ratio of charged Σ baryons to uncharged $\Lambda^0 + \Sigma^0$ cannot be calculated directly because charged Σ baryons cannot be measured with the FOPI detector. Exploiting strangeness conservation and neglecting rare strange particles, the charged Σ yield can be estimated by

$$\Sigma^+ + \Sigma^- = K^0 + K^+ - 2 \cdot K^- - (\Lambda^0 + \Sigma^0) \ . \tag{8.2}$$

Those particles constitute the 'bulk' strangeness produced in the heavy-ion collisions. The K^- is taken into account twice in order to consider the $\overline{K}^0$ yield, which is expected to have approximately the same magnitude. Using total yields of 0.074 ± 0.002 for the K^+ (fit to combined KaoS and FOPI results [49, 50]), 0.003 ± 0.001 for the K^- from KaoS [49] (scaled for the different centrality selection of FOPI [48]) and those of K^0 and $\Lambda^0 + \Sigma^0$ from this work, one calculates a charged Σ yield of 0.022 ± 0.006. Using this value, a $(\Sigma^+ + \Sigma^-)/(\Lambda^0 + \Sigma^0)$ ratio of 0.183 ± 0.053 is determined. This is significantly lower than any of the model predictions.

All models succeed in describing the kaon ratios. When taking into account strange baryons, differences between data and model predictions occur which are most pronounced in the case of the strange baryon ratio. This indicates that all models have problems in properly describing the baryon yields.

Chapter 9

Ξ^- and $\Sigma^-(1385)$: Status

The primary aim of the 2003 experiment was to look for rare strange particles like Ξ^- and $\Sigma^-(1385)$. Taking into account the results of measurements at higher energies, the study of Ξ^- production at SIS energies can contribute to the understanding of strangeness enhancement mechanisms. An investigation of the $\Sigma^-(1385)$ can answer the question to which degree higher resonances of strange baryons are produced in heavy-ion collisions at those energies.

In the following, peculiarities of the analysis are explained and the latest status of the search for Ξ^- and $\Sigma^-(1385)$ is given.

9.1 Challenges in the Analysis

Although reconstruction and analysis procedure for Ξ^- and $\Sigma^-(1385)$ are very similar to that of K^0 and Λ^0, several points have to be considered thoroughly in this case.

Low Production Yields

The production thresholds of Ξ^- and $\Sigma^-(1385)$ in free nucleon-nucleon collisions are 3.74 GeV and 2.39 GeV, respectively. Therefore, at a beam energy of 1.93 AGeV, these particles are produced deeply below the threshold and the yields are expected to be very low. The prediction for the Ξ^-/K^+ ratio by a statistical model for the proposal of the experiment was about $8.5{\cdot}10^{-4}$ in comparison to a measured value of about 1.6 for the $(\Lambda^0+\Sigma^0)/K^0$ ratio ($K^0/K^+ \approx 1$) given in chapter 8.

As a consequence of the low production yields the signal-to-background ratio in the invariant mass spectrum is very small and a possible peak will be hard to identify. This is particularly problematic in the case of the strongly decaying $\Sigma^-(1385)$.

Reconstruction and Cut Strategy

As explained in chapter 5 the reconstruction of both, Ξ^- and $\Sigma^-(1385)$ proceeds in two steps. The first step is to find the Λ^0 by combining proton and π^- tracks. In a second step, this Λ^0 is combined with another π^-.

The selection criteria for the decay particles of Ξ^- and Λ^0 are quite different. The Ξ^- is a weakly decaying particle which implies that a certain fraction of the particles may produce a secondary vertex. Consequently, Λ^0 and π^- from these decays have a finite distance d_0 to the primary vertex. Therefore, they can be distinguished – at least to a certain extent – from primary Λ^0 and π^-. This is not possible for the $\Sigma^-(1385)$ which decays strongly. Here, Λ^0 and π^- come directly from the primary vertex and cannot be separated from primary particles.

In both cases, the signal-to-background ratio in the invariant mass spectra gets worse due to the two-step reconstruction and due to the fact that the π^- from the first decay is emitted at or close to the primary vertex.

Influence of Primarily Produced Λ^0

Λ^0 (and Σ^0) which are directly produced in a collision constitute a source of background for the reconstruction of the $\Sigma^-(1385)$. For the Ξ^- reconstruction, primary Λ^0 can be suppressed by a minimum threshold on d_0 but the π^- stemming from their decay can be mistaken for a pion from the Ξ^- decay. If, for example, π_1^- and p_1 originate from the decay of a real Λ^0, combinations of the type $((\pi_x^-, p_1), \pi_1^-)$ can be produced in the Ξ^- reconstruction. In those cases, (π_x^-, p_1) denotes the Λ^0 which was reconstructed in the first step using the original proton p_1 but another pion p_x. Combining this Λ^0 with pion π_1^- in the second step of the reconstruction will cause an excess in the invariant mass spectrum at a position of about 1.29 GeV/c^2 due to the decay kinematics of the Λ^0. This effect is partly remedied by only selecting events where the number of reconstructed Λ^0 candidates is smaller than 2.

Event Mixing

Describing the combinatorial background with the event-mixing method is a difficult task, especially in the Ξ^- analysis. The presence of primary Λ^0 in the measured events requires a careful selection of events used for mixing. If pions which originate from a real Λ^0 are also used for mixing, an excess around the nominal Ξ^- mass is observed in the mixed-events mass spectrum. Therefore, their decay pions have to be tagged in order to reject them in the event mixing.

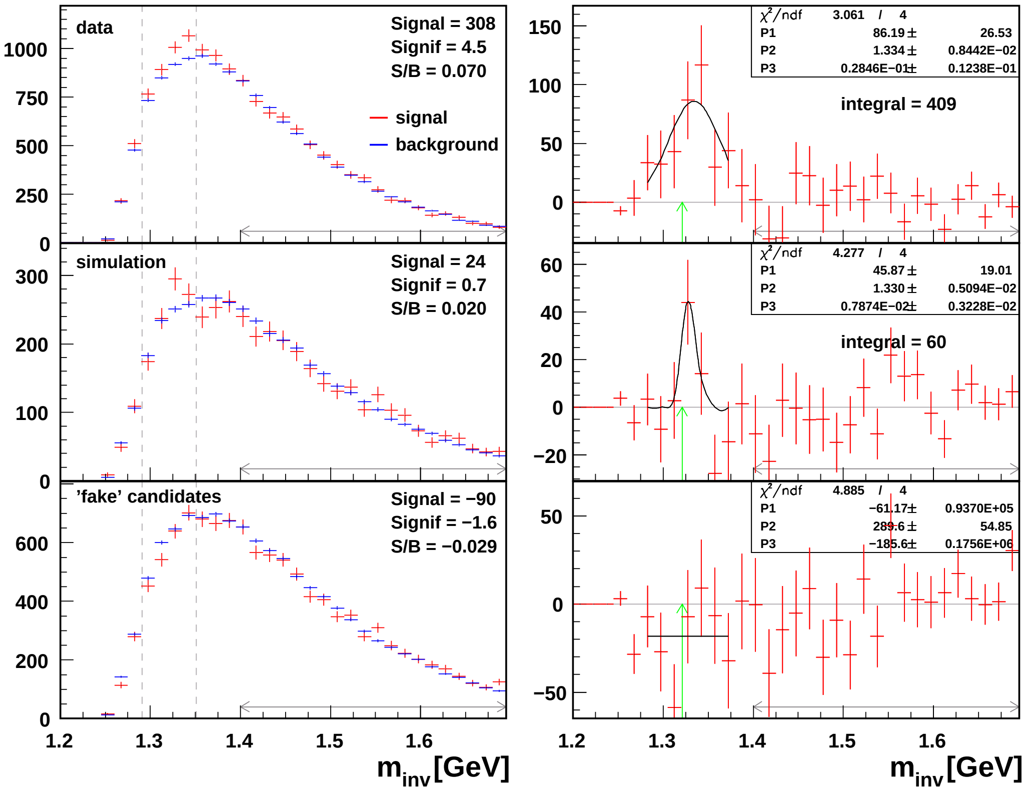


Figure 9.1: Invariant mass spectra of Ξ^- candidates generated from experimental data, from simulated events with embedded thermal Ξ^- and from simulated events with embedded thermal Λ^0 and K^0. The left column shows the combinatorial (red) and mixed-events (blue) distributions, the right column shows the background-corrected mass spectra.

Another issue is the two-track resolution. In the measured data, tracks which are crossing in the active volume of the CDC influence each other causing an efficiency loss. This effect is not present in the event mixing where the tracks are taken from different events. To account for this effect in the reconstruction procedure, strange particle candidates with intersecting pion tracks are taken out from the analysis.

9.2 Analysis Status

An example of reconstructed Ξ^- candidates is given in figure 9.1 for a certain set of selection cuts (cf. appendix E) discarding also crossing π^- tracks. The upper plots show the combinatorial and mixed-events distributions (left) and the background-corrected mass spectrum (right) for the measured data. The middle and lower plots show the corresponding spectra for simulated events with embedded thermal Ξ^- (simulation) and with embedded thermal K^0 and Λ^0 ('fake' candidates), respectively.

The spectrum of the measured data (cf. upper right plot) shows a peak with about 400 candidates (when integrating the Gaussian fit of the peak), a significance of 4.5 and a signal-to-background ratio of 0.07. Using a Gaussian fit, a particle mass of 1.334 MeV/c^2 is determined, shifted by about 13 MeV/c^2 with respect to the nominal Ξ^- mass (indicated by the green arrow). The width of the peak is 28 MeV/c^2.

Various consistency checks have been performed in order to test the authenticity of the peak. The available statistics has been divided into two different data samples and it was verified that the peak is not the result of statistical fluctuations. The distribution of the particle candidates as a function of the run number has been examined to exclude the possibility that they originate from runs which were not calibrated properly. Selecting a different mass window for the Λ^0 proved that the observed signal is only visible if reconstructed Λ^0 with the correct mass are chosen in the analysis. Finally, it was checked that the peak is robust against variations of the analysis cuts.

The analysis of simulated events with embedded thermal Ξ^- showed that with the given set of cuts it is possible to reconstruct these particles (cf. middle right plot). As for the measured data, the peak is shifted towards higher masses (1.330 MeV/c^2) but the resolution is somewhat better ($\sigma = 8$ MeV/c^2) in this case. A reason for this discrepancy has not been found up to now. Therefore, one has to be careful when interpreting the results. Out of 25000 embedded Ξ^-, about 60 are reconstructed which corresponds to an efficiency of 0.24 %. The negative entries left and right from the peak in the corrected mass spectrum indicate that the background is overestimated by the event mixing. When exclusively analyzing thermally generated Ξ^- an efficiency of 0.4 % is determined. In this case the reconstructed peak in the invariant mass spectrum is not influenced by fluctuations due to the background subtraction procedure.

Applying the reconstruction method to simulated events containing thermal Λ^0 and K^0, no signal is detected within the errors. This proves that with this choice of cuts and mixing strategy – even in the presence of primary Λ^0 and K^0 – the analysis method does not create fake Ξ^- candidates and that the reconstruction cuts are able to suppress these primary particles.

In comparison to the Ξ^- analysis the $\Sigma^-(1385)$ reconstruction by itself is less problematic. Due to the decay kinematics the creation of fake candidates by mistaking the pions is not possible. The results are presented in figure

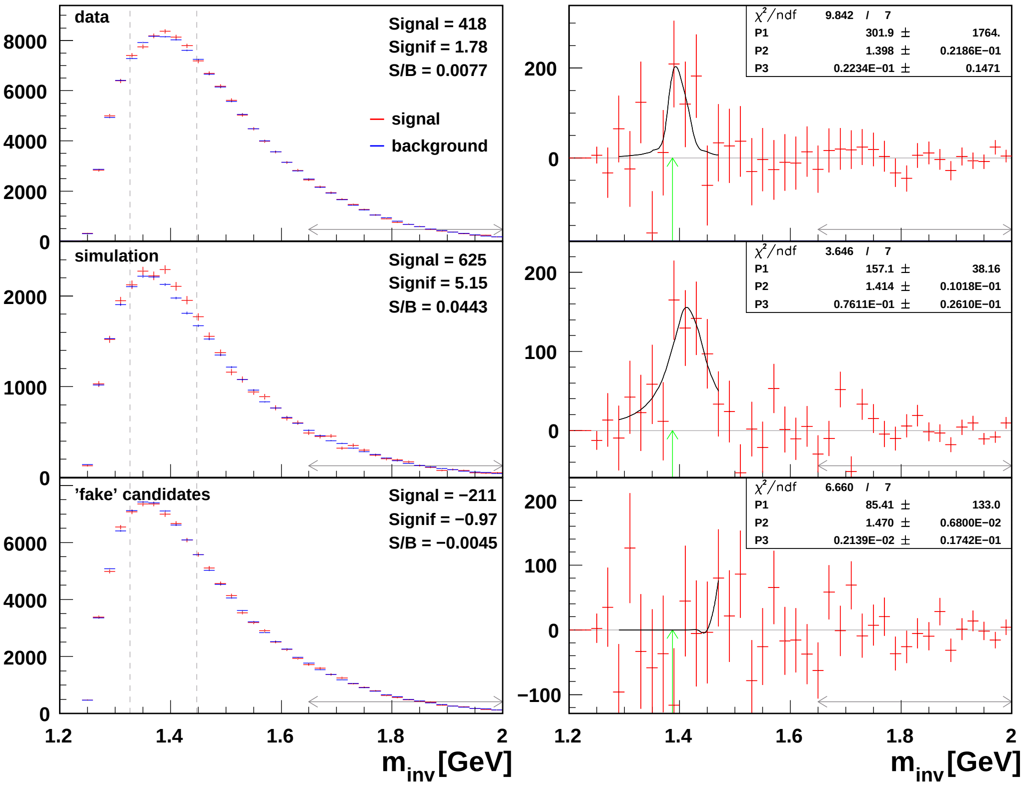


Figure 9.2: Invariant mass spectra of $\Sigma^-(1385)$ candidates generated from experimental data, from simulated events with embedded thermal $\Sigma^-(1385)$ and from simulated events with embedded thermal Λ^0 and K^0. The left column shows the combinatorial (red) and mixed-events (blue) distributions, the right column shows the background-corrected mass spectra.

9.2. In this case the event mixing procedure included upper thresholds for the number of reconstructed Λ^0 and π^- candidates in order to reduce the combinatorial background.

The peaks which are found in the background-corrected mass spectra of measured data and simulated events again are slightly shifted towards higher masses (the nominal $\Sigma^-(1385)$ mass is marked by the green arrow). No fake candidates are reconstructed with the given set of cuts. In the measured data, about 500 ± 200 candidates are found when integrating the fitted function and the efficiency is approximately $2.6{\cdot}10^{-3}$. The signal-to-background ratio is 0.0077 which is roughly a factor ten lower than in the case of the Ξ^-. Therefore, these results will need a verification through a statistics that is an order of magnitude higher.

9.3 Conclusions

As a concluding remark it can be said that the reconstruction method which was developed for the search of cascade particles within FOPI works properly and Ξ^- and $\Sigma^-(1385)$ candidates have been found in the data of the 2003 experiment.

Using the simulation of the detector, an overall Ξ^- reconstruction efficiency of 0.24–0.4 % was determined by analyzing thermally (!) generated particles. Due to the strong cuts which have to be used in order to select secondary Λ^0 and to avoid fake correlations because of a confusion between π^- from the Λ^0 and the Ξ^- decays, this efficiency value is about a factor 7–10 lower than the expectations from the proposal. This implies a 4π yield which is significantly larger than the thermal model prediction – provided that the Ξ^- is emitted isotropically in the center-of-mass system. Since this discrepancy is rather dramatic and potentially very important, the results will have to be verified with significantly larger statistics.

While the $\Sigma^-(1385)$ reconstruction efficiency and the significance of the peak in the invariant mass spectrum have approximately the same size than those of the Ξ^-, the signal-to-background ratio is about a factor ten smaller. For this reason and due to its width the experimental errors on the particle yield are rather large. As for the Ξ^-, an investigation with larger statistics will reduce the errors and will contribute to the understanding of the reconstruction efficiencies and the peak shifts in the invariant mass spectra for both particles.

Chapter 10

Summary and Outlook

In the course of this work the production of strange particles was studied in Ni+Ni collisions at a beam energy of 1.93 AGeV using the FOPI detector. A high-statistics experiment with an upgraded data acquisition system was performed and the resulting data were analyzed for neutral strange particles (Λ^0, K^0) and cascades (Ξ^-, $\Sigma^-(1385)$).

At beam energies of 2 AGeV these particles are produced below or close to the free nucleon-nucleon threshold. For that reason they are sensitive probes for processes affecting their properties. One drawback of the production close to threshold are low particle yields of the order of 0.1 per central event (Λ^0, K^0, K^+) of which only 1 in 20 is typically detectable. In the absence of clear trigger signatures huge numbers of events have to be recorded in order to collect reasonable statistics. Neutral strange particles or cascades can only be reconstructed via their decay products which are measured in the CDC. Its position resolution of particle tracks in the transverse plane allows to identify the (secondary) decay vertices of strange particles. When reading out all wires of this detector the raw data of one event have a size of approximately 1 MB. Transferring this amount of data in FOPI would normally prevent high data-taking rates. By using an efficient online data reduction this problem can be solved.

One part of this work was the implementation of an existing data reduction algorithm on a new hardware component, the SAM3 VME module. Its key component, a digital signal processor, allows for a high-speed parallel processing of the drift chamber data. To exploit all features of the processor in the best possible way, substantial changes had to be made to the algorithm. Furthermore, program parts for recognizing faulty data and handling of errors were added. As a result of these efforts about 110 million events could be recorded within a beam time of about 12 days – an improvement by a factor of ten compared to previous experiments.

The analysis of the neutral strange particles (K^0, Λ^0) produced several interesting results concerning their production and flow. Exploiting the symmetry of the reaction the full phase space was covered.

The transverse mass spectra and the slope parameters of both particles showed that they essentially populate the phase space. Differential yield distributions were determined by integrating the transverse mass spectra and were used to extract the total yield and the width of these distributions. Comparing K^0 slope parameter and yield distributions to K^+ results from a previous experiment, good agreement was found. This leads to the conclusion that inside nuclear matter, production mechanisms are the same for both particles and that isospin differences are unimportant in this case. The corresponding Λ^0 results were confronted with existing proton data measured in the same system at the same beam energy. Here, the situation is found to be different: While the slope parameter distributions agree, the protons exhibit a much broader yield distribution than the Λ^0. An explanation for this observation is that for the Ni system even in central collisions there is no full stopping and that due to the s quark content the Λ^0s have lost memory of their initial movement.

By comparing to calculations of transport models, it was found that the K^0 is described reasonably well by the IQMD model which takes into account in-medium effects for the kaons. The situation for the Λ^0 is different because none of the models is able to describe all measured kinematic observables.

For the first time it was possible to investigate the azimuthal emission of K^0 and Λ^0 over the full rapidity range. The integrated directed flow of the K^0 was found to be compatible with zero which agrees with prior results found for the K^+. Furthermore, the p_t-dependence of the K^0 flow shows the same trend like the K^+ although the errors are large due to the shifted target position. The Λ^0 shows a pronounced flow signal which is equal to that of the protons. This confirms earlier FOPI results with a better accuracy and sheds light on a discrepancy with results from AGS experiments. While the observed Λ^0 flow in Ni+Ni within the errors is fully compatible with our previously published data and with the results of the E810 experiment, the new data clearly indicate a difference to the Au+Au measurements of E895. Further support for our observation comes from the momentum dependence of Λ^0 and proton flow which agrees down to low transverse momenta.

Comparing particle ratios between measured data and model calculations shows that strange mesons are described well by the models, while the predictions for strange baryons differ from the experimental findings. Now, that the bulk part of the produced strange particles has been measured with FOPI it can be – and has to be – verified whether any model is able to consistently describe the experimental results within one scenario. Currently, none of the default versions of the models is able to describe all available data.

The measured data were searched for cascade particles with a newly developed method which is an extension of the neutral strange particle analysis. It became clear that even with the high-statistics data sample, the analysis is extremely challenging. One reason is that despite acceptance optimization, Ξ^- and Σ^- – if they are produced at SIS energies – are ultra-rare particles (one reconstructed particle in about 10^6 events). In part this is also a consequence of the strong cuts which have to be applied in order to distinguish between primary and secondary Λ^0. Furthermore, for the proper determination of the background precautions have to be taken in order to exclude pions originating from primary Λ^0 decays.

Evidence for about 400 Ξ^- candidates was found at an efficiency of approximately 0.4 %. Therefore, the extrapolated 4π yield is significantly larger than any thermal model prediction. A similar number of $\Sigma^-(1385)$ has been reconstructed at approximately the same reconstruction efficiency. Thus, for the first time it was possible to detect this short-lived strange resonance at SIS energies. The present data sample is clearly not sufficient for an unquestionable understanding of the combinatorial background and the properties of the detected peaks in the invariant mass spectra. A data sample which is about ten times larger than the present one will be needed to verify the experimental findings. This will be possible with a future enhancement of the data acquisition.

Future activities of the FOPI experiment in the field of relativistic heavy-ion collisions are currently being prepared. On the detector side, a replacement of the Barrel scintillators by RPC detectors is planned and various prototypes have been tested successfully. This sub-detector will substantially increase the acceptance for charged kaons.

An improved version of the SAM3 module, the SAM4, has already been developed at GSI. It provides a new DSP type which will increase the performance of the data reduction algorithm by about a factor 4-6. It was demonstrated that the code which was developed in the course of this thesis runs also on the new DSP. Therefore, computing speed will no longer be the main issue for the data reduction but other aspects will become more important. The much larger internal RAM (1 MB) of the new DSP can be used to buffer several events thus reducing the overall dead time of the DAQ setup. In addition, there is room for improvement of the reduction algorithm itself:

The peak finding, integration and corrections for individual channels can be extended.

Upcoming experiments will profit from the new DAQ setup. An investigation of strangeness production in π^--nucleus collisions was recently performed and other experiments exploiting the techniques presented in this thesis are currently being planned. They are based on the interesting speculation that due to the strong K^--N interaction, bound states of K^- and nucleons could be formed. Similar to Ξ^- and $\Sigma^-(1385)$ they are detected by analyzing correlations between Λ^0 and protons or deuterons. Thus, FOPI will be able to make important contributions to the understanding of nuclear matter properties in the future.

Appendix A

Models

Drawing conclusions from experimental results mostly involves the comparison to model predictions. Depending on the theoretical approach these models describe different aspects of heavy-ion collisions. The dynamical evolution of a collision can be studied with transport models while statistical models describe the thermodynamical aspects assuming an equilibrium. This appendix gives a brief introduction on both types of models.

A.1 Transport models

Transport models describe nucleus-nucleus collisions in a fully microscopic way. The evolution of the collision is determined in small time intervals taking into account scattering cross sections, particle production and various potentials. To which extent this is done depends on the actual model. In addition, experimental knowledge about cross sections near threshold is scarce and parameterizations have to be used. Models widely used in the SIS energy regime are the BUU models or different types of QMD models, e.g. IQMD and UrQMD:

The BUU model [111] uses the Boltzmann-Uehling-Uhlenbeck transport equation describing collisions under the influence of a mean field. The field, which is a consequence of the surrounding nuclear matter, is can be given in the so-called Skyrme parameterization

$$U(\rho) = A \cdot \left(\frac{\rho}{\rho_0}\right) + B \cdot \left(\frac{\rho}{\rho_0}\right)^{\sigma} \tag{A.1}$$

where $\sigma > 1$, $A < 0$ (attractive part) and $B > 0$ (repulsive part). The non-relativistic BUU equation which has to be solved for the particles in the collision is

$$\frac{\partial f}{\partial t} + \vec{v} \cdot \vec{\nabla}_r f - \vec{\nabla}_r U \cdot \vec{\nabla}_p f = -\frac{1}{(2\pi)^6} \int d^3p_2 d^3p_{2'} d\Omega \frac{d\sigma}{d\Omega} v_{12}$$
$$\times \{[f f_2 (1 - f_{1'})(1 - f_{2'}) - f_{1'} f_{2'} (1 - f)(1 - f_2)]$$
$$\times (2\pi)^3 \delta^3(\vec{p} + \vec{p}_2 - \vec{p}_{1'} - \vec{p}_{2'})\} \ . \tag{A.2}$$

The equation describes the time evolution of the single particle phase-space density $f = f(\vec{r}, \vec{p}, t)$ within a mean field U. The evolution is determined by two-body collisions combined in the collision integral on the right-hand side. The quantity v_{12} is the relative velocity of the colliding particles and the term $\delta^3(\vec{p} + \vec{p}_2 - \vec{p}_{1'} - \vec{p}_{2'})$ ensures the momentum conservation. The Pauli principle is implemented by the [... − ...] term including the one-body distribution functions f for the incoming and outgoing particles. The first part $(f f_2 (1 - f_{1'})(1 - f_{2'}))$ describes the scattering from the considered state into the final state 1',2' and the second part $(f_{1'} f_{2'} (1 - f)(1 - f_2))$ describes the scattering from the state 1',2' into the considered state.

The BUU equation is solved numerically by using a Monte-Carlo method employing test particles. After initialization, the phase-space density f is given by an ensemble of $n(A_P + A_T)$ point-like test particles (A_P and A_T are the numbers of nucleons in the projectile and target nuclei). These particles move classically according to Hamilton's equations:

$$\dot{\vec{p}_i} = -\vec{\nabla}_r U(\rho(\vec{r}_i)) \ , \quad \dot{\vec{r}_i} = \vec{v}_i \tag{A.3}$$

where $\vec{v}_i$ can be calculated relativistically or non-relativistically. The RBUU model, a relativistically correct version, is introduced in [52].

Compared to the one-body description mentioned before, the Quantum Molecular Dynamics (QMD) model [112, 51] is an n-body model which allows to simulate heavy-ion collisions on an event by event basis and which is able to describe the formation of heavy fragments. Each nucleon is described by a function

$$\phi_i(\vec{x}_i, t) = \left(\frac{2}{L\pi}\right)^{3/4} \exp\left\{-(\vec{x}_i - \vec{r}_i(t))^2/L\right\} \cdot \exp\left\{i\vec{x}_i\vec{p}_i(t)\right\} \tag{A.4}$$

with six time-dependent parameters $\vec{r}_i$ and $\vec{p}_i$ and a fixed parameter L describing the extension of the wave packet in phase space. The total n-body wave function is the direct product of the single nucleon wave functions:

$$\Phi(\vec{x}_i, t) = \prod_i \phi(\vec{x}_i, \vec{r}_i, \vec{p}_i, t) \tag{A.5}$$

The aim of this approach, which was developed after BUU, is to describe all particle correlations. However, quantum correlations originating from the antisymmetrization of the wave function $\Phi(\vec{x}_i, t)$ are not taken into account.

The distribution function f_i for the i-th nucleon is derived from the wave function by means of a Wigner transformation,

$$f_i(\vec{r}, \vec{p}, t) = \frac{1}{\pi^3 \hbar^3} \exp\left\{ -\frac{2}{L}(\vec{r} - \vec{r}_i(t))^2 - \frac{L}{2\hbar^2}(\vec{p} - \vec{p}_i(t))^2 \right\} , \tag{A.6}$$

and for the expectation value of the Hamiltonian one finds

$$\langle H \rangle = \sum_i \frac{p_i^2}{2m_i} + \sum_i \sum_{j>i} \int f_i(\vec{r}, \vec{p}, t) V^{ij} f_j(\vec{r}', \vec{p}', t) d\vec{r} d\vec{r}' d\vec{p} d\vec{p}' \; . \tag{A.7}$$

The baryon potential V^{ij} has different contributions like a Skyrme-type and a Yukawa potential, a Coulomb potential and momentum dependent interactions. The equations of motion are given by

$$\dot{\vec{p}_i} = -\frac{\partial \langle H \rangle}{\partial \vec{r}_i} \; , \quad \dot{\vec{r}_i} = -\frac{\partial \langle H \rangle}{\partial \vec{p}_i} \tag{A.8}$$

Two different realizations of the QMD model are used for comparison with the experimental results. The IQMD model [51] explicitly takes into account the isospin dependence of the nucleon potential and the cross sections. It was developed for the SIS energy range and is also used to generate particle distributions for the simulation of the FOPI detector (cf. chapter 6). The UrQMD model [109] describes ultra-relativistic heavy-ion collisions at energies reached at the AGS or the SPS. About 50 different baryon species like nucleons, deltas, hyperons and their resonances (with masses up to 2.11 GeV/c^2) and 25 meson species are taken into account in the model.

A.2 Statistical Models

Statistical models [113, 114] are an approach to explain particle production (yields, ratios) in heavy ion collisions assuming thermal and chemical equilibrium. In this framework, two parameters are sufficient to describe all particle yields, the temperature T and the baryonic chemical potential μ_B. The formalism of the grand canonical ensemble is employed to predict particle ratios for a given set of the model parameters. Assuming an ideal gas the density n_i of a particle species i is described by

$$n_i = \frac{N_i}{V} = \frac{g_i}{2\pi^2} \int_0^\infty \left[\exp\left(\frac{E_i(p) - \mu_i}{T} \right) + \varepsilon \right] p^2 dp \; , \tag{A.9}$$

where N_i is the total number of particles of species i, V is the volume of the hadronic system at freeze-out, $E_i(p) = \sqrt{p_i^2 + m_i^2}$ is the particle energy, g_i is the spin degeneracy factor and $\varepsilon = \pm 1$ for fermions or bosons. The chemical potential μ_i can be decomposed into

$$\mu_i = \mu_B B_i - \mu_S S_i - \mu_{I_3} I_{3i} \tag{A.10}$$

with the baryon, strangeness and isospin (3^{rd} component) quantum numbers B_i, S_i and I_{3i} and the corresponding potentials. The quantum numbers are fixed with the following conservation laws:

$$\begin{array}{ll}
\textbf{Baryon number} & V\sum_i n_i B_i = Z + N \\
\textbf{Strangeness} & V\sum_i n_i S_i = 0 \\
\textbf{Isospin} & V\sum_i n_i I_{3i} = \dfrac{Z - N}{2}
\end{array}$$

Here, Z and N are the total proton and neutron numbers in the colliding system. The volume V is eliminated by regarding particle ratios.

At SIS/GSI energies, strange particles are produced near or below the production threshold and the freeze-out temperatures are below 100 MeV. Therefore, the number of strange particles produced in a collision is small. Instead of the grand canonical ensemble, a canonical description using the exact conservation of strangeness should be employed [114, 57].

Appendix B

The Bethe-Bloch Formula

The energy loss of a charged particle crossing a material medium is described by the Bethe-Bloch formula [74]:

$$-\frac{dE}{dx} = 2\pi N_a r_e^2 m_e c^2 \rho \frac{Z}{A} \frac{z^2}{\beta^2} \left[\ln \left(\frac{2 m_e \gamma^2 v^2 W_{max}}{I^2} \right) - 2\beta^2 \right] \qquad \text{(B.1)}$$

with the following constants, parameters and variables

N_a: Avogadro's number: $6.022 \cdot 10^{23}$ mol^{-1}
r_e: class. electron radius: 2.818 fm
$m_e c^2$: electron mass: 511 keV

Z: atomic number of absorber
ρ: density of absorber
β: v/c of incident particle
A: atomic weight of absorber
z: charge of incident particle
γ: $1/\sqrt{1-\beta^2}$

I: mean ionization potential: ≈ 10 eV
W_{max}: maximum energy transfer in single collision

For large masses of the incident particle ($M \gg m_e$) the maximum energy transfer for a single collision can be approximated by $W_{max} \approx 2 m_e c^2 (\gamma^2 \beta^2)$. Thus the Bethe-Bloch formula can be rewritten as

$$-\frac{dE}{dx} = 4\pi N_a r_e^2 m_e c^2 \rho \frac{Z}{A} \frac{z^2}{\beta^2} \left[\ln \left(\frac{2 m_e c^2 \gamma^2 \beta^2}{I} \right) - \beta^2 \right] . \qquad \text{(B.2)}$$

For low velocities the logarithmic term varies only slowly and β^2 is negligible. Therefore the energy loss $-dE/dx$ is inversely proportional to β in first order:

$$-\frac{dE}{dx} \propto \frac{z^2}{\beta^2} \ . \tag{B.3}$$

Taking into account the dE/dx information from the CDC and the velocity measured with the Barrel the particle charge can be determined with the above relation.

If only the CDC information about dE/dx and p/z is available the particle mass can only be determined if the charge is assumed to be 1, 2 or any higher integer value:

$$-\frac{dE}{dx} \propto \frac{z^2}{\beta^2} = z^2 \cdot \left(\frac{\gamma^2}{\gamma^2 - 1}\right) = z^2 \cdot \left(1 + \frac{1}{\gamma^2 - 1}\right) = z^2 + \frac{m^2}{(p/z)^2} \ . \tag{B.4}$$

Appendix C

Kinematic Variables

In heavy-ion physics it is desirable to have kinematic variables which allow for a simple change of the reference frame, e.g. between the laboratory frame and the center-of-mass frame (for a more detailed description see e.g. [115], chapter 2).

The rapidity y describes the longitudinal velocity β_z ($= v_z/c$) of a particle with total energy E and longitudinal momentum p_z:

$$y = \frac{1}{2} \ln \left(\frac{E + p_z}{E - p_z} \right) = \tanh^{-1} \beta_z \tag{C.1}$$

Although this variable depends on the reference frame, it shows a very convenient behaviour under a Lorentz transformation. If, for example, a rapidity y measured in the laboratory frame has to be transformed into a rapidity y' in the center-of-mass frame which is moving with a velocity β_{cm} with respect to the laboratory frame, the relation between y and y' is simple:

$$y' = y - \frac{1}{2} \ln \left(\frac{1 + \beta_{cm}}{1 - \beta_{cm}} \right) = y - y_{cm} \tag{C.2}$$

To be able to compare between different beam energies the scaled rapidity y^0 is used:

$$y^0 = \frac{y}{y_{cm}} - 1 \quad \left(= \frac{y - y_{cm}}{y_{cm}} \right) \tag{C.3}$$

This is the rapidity in the center-of-mass frame ($y - y_{cm}$) normalized to the center-of-mass rapidity y_{cm}. If the collision system is mass symmetric the target-, mid- and projectile-rapidity values are -1, 0 and +1, respectively.

The transverse momentum p_t is invariant under a Lorentz transformation along the beam direction:

$$p_t = \sqrt{p_x^2 + p_y^2} \quad \left(= \beta_t \gamma m_0 = \frac{\beta_t}{\sqrt{1 - \beta^2}} m_0 \right) \tag{C.4}$$

with β_t being the transverse component of β ($=v/c$). Using p_t and the rest mass m_0 of a certain particle its transverse mass can be defined:

$$m_t = \sqrt{p_t^2 + m_0^2} \tag{C.5}$$

The motivation for its use is the simple relation with the energy E and the longitudinal momentum p_z:

$$E = m_t \cdot \cosh(y) \tag{C.6}$$

$$p_z = m_t \cdot \sinh(y) \tag{C.7}$$

To calculate the center-of-mass rapidity y_{cm} for a mass symmetric collision it is sufficient to determine the total energy E_N and the momentum p_N of a single nucleon inside the projectile nucleus neglecting the Fermi momentum:

$$E_N = E_{beam} + m_{amu} \tag{C.8}$$

$$p_N = \sqrt{E_N^2 - m_{amu}^2} \tag{C.9}$$

where E_{beam} is the kinetic energy of the beam (1.93 AGeV) and m_{amu} is the atomic mass unit (0.9315 GeV/c^2). Then the center-of-mass rapidity is

$$y_{cm} = \frac{y_N}{2} = \frac{1}{4} \ln \left(\frac{E_N + p_N}{E_N - p_N} \right) \tag{C.10}$$

which, for the given beam energy, has a value of 0.894.

Appendix D

The DSP Data Reduction Code

This chapter will provide a short documentation of the DSP data reduction code. First, the contents of the DSP code directory are reported and available programs are introduced briefly. The code of the data reduction program used in the S261 experiment is presented afterwards and short descriptions of all subprograms are given. In the end, the procedure of generating an executable program for the DSP is explained.

D.1 Source-Code Directory Content

An overview of the structure and the content of the DSP code directory is given in figure D.1. Directories starting with 'x_' contain the main source code file (.c) of a DSP program and a makefile (.mak) which is needed for the compilation. Shared subprograms are located in the 'src' subdirectory where they are sorted according to their function. Include files (.h) for all programs are found in the 'inc' subdirectory and libraries (.lib) providing special functions for the DSP are stored in the 'lib' subdirectory. The 'cmd' subdirectory contains setup files which define the partitioning of the DSP internal RAM for the compiler and the final programs (.m0) are placed in 'm0_files'.

D.2 Available DSP Programs

This section will provide a short description of all available DSP programs located in the 'm0_files' directory.

- **dmatest.m0:** This program was used to check whether the DMA transfer between different RAMs works properly. Source and desti-

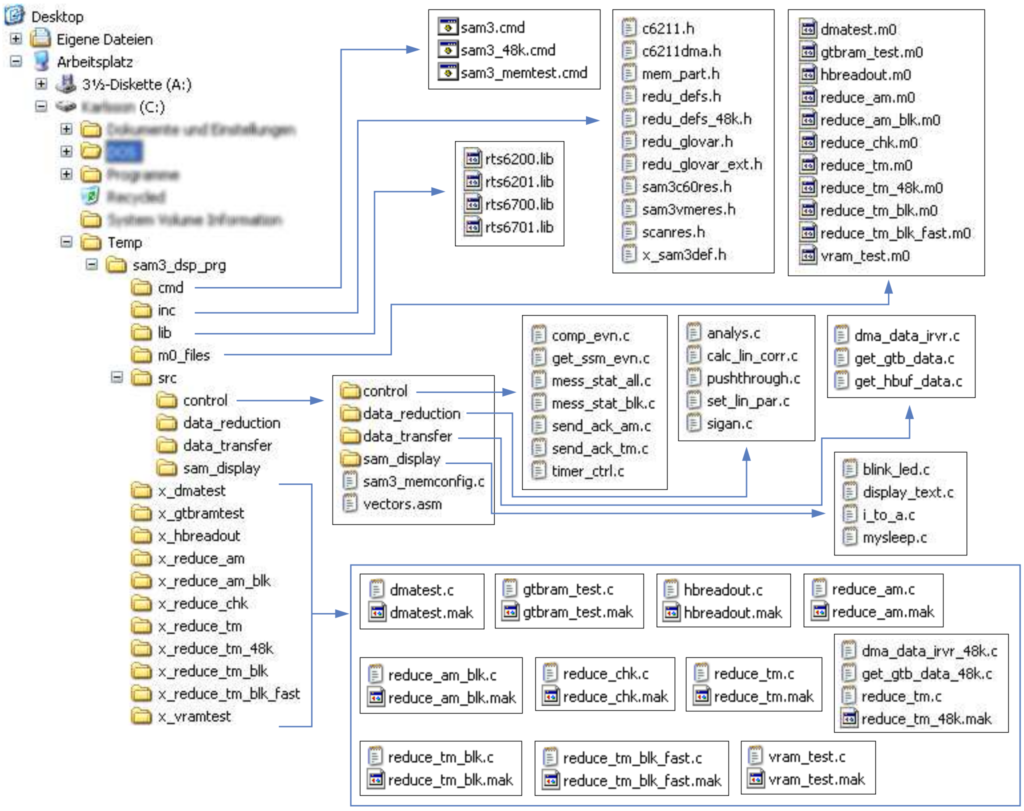


Figure D.1: Structure and content of the DSP code directories. Program directories start with 'x_', the other directories contain include files, libraries or shared subprograms.

nation of the transfer can be chosen among GTBRAM, DSP internal RAM and VRAM. The source memory is filled with various patterns of a given length, the DMA control registers are prepared and the transfer is initiated. Afterwards, the data in source and destination RAM are compared and errors are counted. The duration of the transfer can also be determined.

- **gtbram_test.m0:** Testing read/write accesses of the GTBRAM was done with this program. First, according to a certain seed number, random bit patterns were written into the RAM. Thereafter, its content was read and compared to the regenerated random bit pattern while counting possible errors.

- **vram_test.m0:** This program works in the same way as the test of the GTBRAM. Here, the random bit patters are written to the VRAM in order to check for read/write errors in the data transmission.

- **hbreadout.m0:** The communication between SSM, DSP and RIO and the readout of the FADC data stored in the SSM's hit-buffer memory was first exercised with this routine. The data from the hit-buffer are read in the so-called 'address mode'.
- **reduce_chk.m0:** This is a combination of the various tests which were described above. In addition, the hit-buffer content can be compared to the GTBRAM content to test the data transmission between SSM and SAM in the 'token mode'.
- **reduce_am.m0:** The first fully functional data reduction program was realized here. First consistency checks (event number, hit-buffer data length) are performed and different program modes for data reduction, copying of raw data and filtering of valid hitlets are available. The data transfer between SSM and SAM is done in the address mode.
- **reduce_am_blk.m0:** Works like the prior program with the exception that GTBRAM and VRAM each are divided into two separate blocks buffering incoming and outgoing data. Thus the dead time of the DAQ system is reduced.
- **reduce_tm.m0:** The program is similar to 'reduce_am' but instead of the address mode it uses the token mode for transferring the SSM data into the GTBRAM.
- **reduce_tm_48k.m0:** In order to allow for the readout and reduction of non-zero-suppressed raw data this special version of 'reduce_tm' was created. Due to the small program size, 48 kB of the internal RAM of the DSP remain free for data which is sufficient for reading approximately 500 (= 5 μs) FADC samples per wire.
- **reduce_tm_blk.m0:** Works like 'reduce_tm' with the additional buffering of data in GTBRAM and VRAM.
- **reduce_tm_blk_fast.m0:** This is the fastest available DSP program for data reduction. The option to copy raw data was dropped to decrease the size of the code. Therefore, together with the zero-suppression by the SSM a scan range of about 600 samples is possible.

Many of those programs were created for testing purposes only. The ones used in the experiment are '**reduce_tm_48k**' for the direct readout of non-zero-suppressed raw data and '**reduce_tm_blk_fast**' for the standard operation of the DAQ system. A detailed explanation of the latter one will be given in the following section.

D.3 Example Program Code

To get insight into the structure of a DSP program and to provide some documentation, the source code of the **reduce_tm_blk_fast**' program is presented in this section. First, the program structure is described. The include files containing hardware addresses, program parameters and global variables are provided afterwards. Subprograms for the initialization of memories and linearization correction, for the data reduction and for the data transfer are explained.

D.3.1 Main Program

The general structure of a data reduction program was already described in table 3.1 on page 40. Here, the source code of the DSP program used in the experiment is presented.

After getting hardware addresses and program parameters from the include files (lines 13–22), all function prototypes are listed (lines 29–48) and some variables are defined (lines 54–70). The program option and the number of pre- and post-samples for the pedestal correction are read from a region in the VRAM which is used for the communication between RIO and DSP (defined in `mem_part.h`) and the memory address which is used to monitor the DSP program status is cleared. Due to hardware reasons the number of post-samples is ignored (lines 78–84).

Before starting normal operation, the program status is checked, the external memory (GTBRAM and VRAM) properties are configured for the EMIF and the lookup tables for the FADC linearization correction are prepared. The base addresses of the memory blocks in the VRAM which will contain the processed data are calculated and communicated to the RIO. Furthermore, the SSM is started, the status flags of the memory blocks are set and the DSP event number is cleared (lines 89–134).

At the beginning of each cycle of the following loop the DSP event number is increased and the active memory block is selected. By checking its status flag it is determined whether the RIO has already retrieved the data of a previous event which was stored there. If the block is free, its counter for the data length is reset and the DSP event number of the coming event is stored (lines 138–157). The status of the SSM is checked by looking at its status register. If scanning was finished and the FADC data was written into the

GTBRAM, the SSM is stopped and its event number is also written to the corresponding data block in the VRAM. DSP and SSM event numbers are compared and, in case of disagreement, an error message is sent to the RIO and the processing of the event is canceled (lines 162–191). After a successful comparison, the size of the event is determined. If the event is too big to fit into the internal RAM of the DSP or if the SSM sent no data at all, an error is reported to the RIO. In addition, the integrity of the event is checked by looking at two status words at the end of the data (lines 196–240).

If no error occurred up to this point, the data are copied from the GTBRAM to the internal RAM and the SSM is prepared for the scanning of the next event (lines 246–253). The data reduction procedure is performed and, in case of a successful reduction, the results will be in the VRAM where they can be retrieved by the RIO (lines 272–312).

```
 1 //
 2 // read out hit buffer (SSM) in address mode,
 3 //  reduce data
 4 //  write data to VMERAM and notify RIO
 5 //
 6 //    use only (!) in RED mode
 7 //
 8
 9 //  |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
10 //       10        20        30        40        50        60        70        80        90      100
11
12
13 #include "c6211.h"                            // c6x11 global definitions
14 #include "c6211dma.h"                         // c6x11 DMA options
15 #include "scanres.h"                          // SSM resources
16 #include "sam3c60res.h"                       // SAM3 DSP resources
17 #include "sam3vmeres.h"                       // SAM3 VME resources
18
19 #include "mem_part.h"                         // DSP/RIO memory partitioning
20
21 #include "redu_defs.h"                        // special definitions f. data reduction
22 #include "redu_glovar.h"                      // global variables
23
24
25
26 // function prototypes
27
28 // *** function prototypes ***
29         void sam3_memconfig();                          // configure SAM3 memories
30
31         void set_lin_par();                             // set linearisation compression parameter
32         void calc_lin_corr();                           // calc. linearisation correction values
33
34 //unsigned int analys(const unsigned short *,          // analyse raw data for reduction
35 //                    const unsigned int *);
36 unsigned int analys(unsigned short *,                   // analyse raw data for reduction
37                     unsigned int *);
38 unsigned int sigan(const unsigned int,                  // reduce raw data
39                    const unsigned int);
40
41 unsigned int get_gtb_data(unsigned int, unsigned int);  // get event data from GTBRAM block
42
43 unsigned int get_ssm_evn(unsigned int);                 // get SSM evt. # from GTBRAM
44 unsigned int timer_ctrl(unsigned int, unsigned int);    // control DSP timer functions
45 unsigned int comp_evn(unsigned int,                     // compare SSM & DSP event numbers
46                           unsigned int, unsigned int);
47         void mess_stat_blk(unsigned int, unsigned int); // write program status word to register
48         void send_ack_tm(unsigned int);                 // set next mem. block & send ’Ack’
49
50
51 void main()
52 {
53
54 //MM           unsigned int    dsp_no;                                // DSP number
55            unsigned int    prg_mod;                               // program mode (-> ’mem_part.h’)
56
57            unsigned int    rio_rdy;                               // RIO ’Rdy’ flag
58            unsigned int    ssm_rdy;                               // SSM ’Rdy’ flag
59
60            unsigned int    dsp_evn;                               // DSP event number
```

```
           unsigned int    ssm_evn;                              // SSM event number

           unsigned int    hb_wcnt;                              // # of 16bit words in hit buffer
           unsigned int    fifo_ovfl;                            // # of SSM FIFO overflows

           unsigned int    i;                                    // loop counter
           unsigned int    tmp1,tmp2;                            // temporary 'int' variables
           unsigned int    d_cnt;                                // written data (byte) counter

           unsigned int    i_blk;                                // active VRAM block index:
                                                                 //   0 : VRAM block 0
                                                                 //   1 : VRAM block 1


// *** get options from VMERAM & clean up comm. reg. ***

//MM  dsp_no   =  *(unsigned int *)DSP_OPT_NUM;                      // get DSP number
  prg_mod  =  *(unsigned int *)DSP_OPT_CP0 & 0xffff;            // get program mode
  pre_smpl = (*(unsigned int *)DSP_OPT_PPS >> 16) & 0xffff;       // get number of presamples
  pst_smpl =  *(unsigned int *)DSP_OPT_PPS & 0xffff;            // get number of postsamples

  *(volatile unsigned int *)DSP_OPT_CTL = COM_RegReset;         // reset comm. register
/////////////
  pst_smpl = 0;                                                  // IGNORE POSTSAMPLES !!!
////////////

// *** check program mode ***

  if (prg_mod != MOD_RED) {
    while (1) {
      *(volatile unsigned int *)DSP_OPT_CTL = COM_Error;
    }
  }


// *** configure SAM3 memories for DSP ***

  sam3_memconfig();                                              // configure memory

  *(volatile unsigned int *)DSP_OPT_CTL = COM_ProgStart;        // comm. reg.: prg. started


// *** prepare linearisation value/parameter table ***

  set_lin_par();                                                 // set up lin. parameters
  calc_lin_corr();                                               // calc. lin. correction


// *** set memory block addresses for RIO (w/o VRAM_OFFS !!!) ***

  for (i=0; i<NBLKS; i++) {
    tmp1 = DSP_OPT_DA0 + i*BOPT_OFFS;                           // i-th blocks start addr. reg.
    tmp2 = RIO_VR1_D0  + i*RBLK_SIZE;                           // i-th blocks start addr. (RIO)
    if (i > 0) {                                                  // 2nd,3rd... block?
      tmp2 -= OPT_LEN;                                           // subtract option block length
    }
    *(volatile unsigned int *)tmp1 = tmp2;                       // save start address (offset)
  }



// *****************************************
// *** start 'get data' loop (main loop) ***
// *****************************************

  send_ack_tm(1);                                                // send 'Ack' to SSM and...
                                                                 // ...GTBRAM cnt. -> blk. 0 !!!
  *(volatile unsigned int *)DSP_OPT_CTL    = COM_MainLoop;      // prep. OK; now do main loop

  *(volatile unsigned int *)(DSP_OPT_ST0) = COM_DataRetrieved;  // initialize block 0
  *(volatile unsigned int *)(DSP_OPT_ST0 + BOPT_OFFS)
                                          = COM_DataRetrieved;   // initialize block 1

  dsp_evn = 0;                                                   // reset event # counter

  while (1) {                                                     // (b) main DSP loop

    dsp_evn += 1;                                                // increase DSP event number
    i_blk = (dsp_evn-1)%2;                                       // calc. current block #


// *** DSP <-> RIO communication: is current VRAM block empty again? ***

    rio_rdy = *(volatile unsigned int *)(DSP_OPT_ST0
                                         + i_blk*BOPT_OFFS);     // look for 'Rdy' from RIO
    while (rio_rdy != (unsigned int)COM_DataRetrieved) {
      rio_rdy = *(volatile unsigned int *)(DSP_OPT_ST0 + i_blk*BOPT_OFFS);
    }

    *(volatile unsigned int *)(DSP_OPT_DL0+i_blk*BOPT_OFFS) = 0; // reset data length register



// *** save DSP event number ***

    *(volatile unsigned int *)(DSP_OPT_DE0
                               + i_blk*BOPT_OFFS) = dsp_evn;     // store DSP evt. #


// *** DSP <-> SSM communication: has SSM finished scanning? ***

    mess_stat_blk(COM_WaitForAck, i_blk);                       // save program status

    ssm_rdy = 0;
    while (ssm_rdy == 0) {
      ssm_rdy = *(volatile unsigned int *)DSP_GTBXDSR & DSR_RDY; // look for 'Rdy'
    }
    *(volatile unsigned int *)DSP_GTBXDSR &= ~DSR_ACK;           // take back 'Ack'

///////////////////////////////////////////////////////
    timer_ctrl(0,0);                                             // reset & start timer 0
///////////////////////////////////////////////////////


// *** get SSM event number ***

    ssm_evn = get_ssm_evn(i_blk);                               // get SSM event #
    *(volatile unsigned int *)(DSP_OPT_SE0
                               + i_blk*BOPT_OFFS) = ssm_evn;     // store SSM evt. #


// *** compare & read back event numbers ***

    tmp1 = 0;
    tmp1 = comp_evn(dsp_evn, ssm_evn, i_blk);                   // start evt. # comparison
    if (tmp1 != 0) {

      send_ack_tm(i_blk);                                       // send 'Ack' to SSM
      mess_stat_blk(tmp1, i_blk);                               // save program status
      goto too_big_evt;
    }


// *** get hitbuffer/GTBRAM content size (in 16bit words, token mode!!!) & # of FIFO overflows ***

    tmp1 = *(volatile unsigned int *)DSP_GTBXADRPTR;             // read GTBRAM counter
    hb_wcnt = (tmp1 & 0xffff) - i_blk*(GBLK_SIZE/L_SHORT);      // get # 16bit words in GTBRAM

    fifo_ovfl = *(volatile unsigned int *)(GTB_DATA_0           // get # of FIFO overflows...
                              + i_blk*(GBLK_SIZE/L_SHORT)*L_INT // ... set block start adress...
                              + (hb_wcnt-2)*(L_INT/L_SHORT));    // ... go to overflow counter


// *** check size of hitbuffer/GTBRAM content ***

    if (hb_wcnt > IBLK_SIZE/L_SHORT) {                           // raw data size > 32kB ???

      send_ack_tm(i_blk);                                       // send 'Ack' to SSM
      mess_stat_blk(COM_Error_BigEvent, i_blk);                // save program status
      goto too_big_evt;
    }

    if (hb_wcnt <= 6) {                                           // ...or no event at all ???

      *(volatile unsigned int *)(DSP_OPT_DL0                     // reset data length
                            + i_blk*BOPT_OFFS) = 0;

      send_ack_tm(i_blk);                                       // send 'Ack' to SSM
      mess_stat_blk(COM_DataRdy, i_blk);                        // save program status
      goto too_big_evt;
    }


// *** check event integrity (2*0xffff at end?) ***

    tmp1 = *(volatile unsigned int *)(GTB_DATA_0                 // get 1st '0xffff'
                            + i_blk*(GBLK_SIZE/L_SHORT)*L_INT
                            + (hb_wcnt-4)*L_INT);
    tmp2 = *(volatile unsigned int *)(GTB_DATA_0                 // get 2nd '0xffff'
                            + i_blk*(GBLK_SIZE/L_SHORT)*L_INT
                            + (hb_wcnt-3)*L_INT);
    tmp1 = tmp1 & 0xffff;
    tmp2 = tmp2 & 0xffff;

    if ((tmp1 != 0xffff) || (tmp2 != 0xffff)) {                 // bad event?

      send_ack_tm(i_blk);                                       // send 'Ack' to SSM
      mess_stat_blk(COM_Error_BadEvent, i_blk);                // save program status
      goto too_big_evt;
    }



// *** get data from GTBRAM ***

    mess_stat_blk(COM_StartToCopy, i_blk);                      // save program status

    d_cnt = get_gtb_data(hb_wcnt,i_blk);                       // get data from GTBRAM


// *** DSP <-> SSM communication: set next 'Ack' ***

    send_ack_tm(i_blk);                                         // send 'Ack' to SSM


// *********************************
// *** go through different modi ***
// *********************************

///////////////////////////////////////////////////////
    timer_ctrl(1,0);                                             // reset & start timer 1
///////////////////////////////////////////////////////

//MM    if ((prg_mod & MOD_RED) > 0) {                               // (b) 'REDUCE' data

//MM      if (prg_mod == MOD_RED) {                                 // exclusively 'REDUCE' ?
        i_blk = (dsp_evn-1)%2;                                  // switch bw. VRAM blocks
//MM      } else {
//MM        i_blk = 0;                                              // always to VRAM block 0
//MM      }

      mess_stat_blk(COM_StartAnalysis, i_blk);                  // save program status

      tmp1 = (DSP_IRAM_D);                                      // raw data location
      tmp2 = (DSP_VME_D0 + i_blk*DBLK_SIZE);                    // results location
      if (i_blk > 0) {
        tmp2 -= OPT_LEN;
      }
      *(volatile unsigned int *)tmp2 = (fifo_ovfl << 16)          // save FIFO overflows
                                     + hb_wcnt*L_SHORT;         // save HB wordcounter
      tmp2 += L_INT;                                            // incr. address

//      d_cnt = analys((const unsigned short *)tmp1,
//                     (const unsigned int *)tmp2);             // reduce data
      d_cnt = analys((unsigned short *)tmp1,
                     (unsigned int *)tmp2);                      // reduce data

//////////////////////////////////////////////////////////////////////////////////////////////////
      if ((d_cnt == COM_Error_BadSLen) ||
          (d_cnt == COM_Error_BadNPk)) {                        // error in 'analys'/'sigan'?
//18.01.2003 (many 0xeff0eff errors during beamtime)
//        mess_stat_blk(COM_Error_BadEvent, i_blk);               // save program status
        mess_stat_blk(d_cnt, i_blk);                            // save program status
        goto too_big_evt;                                       // jump to end
      }
//////////////////////////////////////////////////////////////////////////////////////////////////

///////////////////////////////////////////////////////
      tmp2 = (DSP_VME_D0 + i_blk*DBLK_SIZE);                    // results location
      if (i_blk > 0) {
        tmp2 -= OPT_LEN;
      }
      *(volatile unsigned int *)tmp2  = 0x00000000;
      *(volatile unsigned int *)tmp2 += (timer_ctrl(1,1)/25 << 16); // save timer 1 in [us]
      *(volatile unsigned int *)tmp2 += (timer_ctrl(0,1)/25);    // save timer 0 in [us]
///////////////////////////////////////////////////////

      d_cnt += L_INT;                                           // incr. data length (fifo/hb)
      *(volatile unsigned int *)(DSP_OPT_DL0
                               + i_blk*BOPT_OFFS) = d_cnt;      // save data length
      *(volatile unsigned int *)(DSP_OPT_ST0
                               + i_blk*BOPT_OFFS) = COM_DataRdy; // give ready signal

//MM    }                                                            // (e) 'REDUCE' data



// *** DSP <-> RIO communication: data in VRAM ***

    *(volatile unsigned int *)DSP_OPT_CTL = COM_DataRdy;        // => data ready in VRAM for RIO


    too_big_evt:


// *** DSP communication: starting over with main loop ***
```

```
    *(volatile unsigned int *)DSP_OPT_CTL = COM_MainLoop;       // run main loop again


  }                                                              // (e) main DSP loop


}
```

D.3.2 Include Files

All hardware addresses, program parameters, global variables, error codes and the partitions of the SAM3 memories are described in various include files which are documented in this section.

DSP Resources

Internal hardware addresses of the DSP are defined in the file `c6211.h`. Lines 13-20 contain those of the external memory interface (EMIF) which are needed later on in the setup of the specific SAM3 memories. The two timers of the DSP are controlled by the addresses given in lines 55–62 and the interrupt and control registers for the DMA transfers are set up in lines 65–72. They will be needed in the routines for copying the data from one memory to another one. Lines 103–114 define the addresses of the QDMA registers and pseudo registers. Both types of registers contain the DMA transfer options, the source and destination addresses for the transfer and other information. Setting up a transfer is done by writing all necessary parameters to the normal registers. Afterwards, the transfer is started by writing to one of the pseudo registers, e.g. the destination address. The bit positions of memory types and timing parameters for the EMIF which have to be used for the configuration are provided in lines 120–134.

```
 1 /*******************************************************************************
 2 * FILENAME
 3 *   c6211dsk.h
 4 *
 5 * DESCRIPTION
 6 *   DSK Header File
 7 *
 8 *******************************************************************************/
 9
10 /* Register definitions for C6211 chip on DSK */
11
12 /* Define EMIF Registers  */
13 #define EMIF_GCR         0x1800000    /* Address of EMIF global control       */
14 #define EMIF_CE0        0x1800008    /* Address of EMIF CE0 control           */
15 #define EMIF_CE1        0x1800004    /* Address of EMIF CE1 control           */
16 #define EMIF_CE2        0x1800010    /* Address of EMIF CE2 control           */
17 #define EMIF_CE3        0x1800014    /* Address of EMIF CE3 control           */
18 #define EMIF_SDCTRL        0x1800018    /* Address of EMIF SDRAM control       */
19 #define EMIF_SDRP        0x180001c    /* Address of EMIF SDRM refresh period       */
20 #define EMIF_SDEXT        0x1800020    /* Address of EMIF SDRAM extension       */
21
22 /* Define McBSP0 Registers */
23 #define McBSP0_DRR      0x18c0000   /* Address of data receive reg.        */
24 #define McBSP0_DXR      0x18c0004   /* Address of data transmit reg.       */
25 #define McBSP0_SPCR     0x18c0008   /* Address of serial port contl. reg.  */
26 #define McBSP0_RCR      0x18c000C   /* Address of receive control reg.     */
27 #define McBSP0_XCR      0x18c0010   /* Address of transmit control reg.    */
28 #define McBSP0_SRGR     0x18c0014   /* Address of sample rate generator    */
29 #define McBSP0_MCR      0x18c0018   /* Address of multichannel reg.        */
```

```
#define McBSP0_RCER     0x18c001C   /* Address of receive channel enable.   */
#define McBSP0_XCER     0x18c0020   /* Address of transmit channel enable.  */
#define McBSP0_PCR      0x18c0024   /* Address of pin control reg.          */

/* Define McBSP1 Registers */
#define McBSP1_DRR      0x1900000   /* Address of data receive reg.         */
#define McBSP1_DXR      0x1900004   /* Address of data transmit reg.        */
#define McBSP1_SPCR     0x1900008   /* Address of serial port contl. reg.   */
#define McBSP1_RCR      0x190000C   /* Address of receive control reg.      */
#define McBSP1_XCR      0x1900010   /* Address of transmit control reg.     */
#define McBSP1_SRGR     0x1900014   /* Address of sample rate generator     */
#define McBSP1_MCR      0x1900018   /* Address of multichannel reg.         */
#define McBSP1_RCER     0x190001C   /* Address of receive channel enable.   */
#define McBSP1_XCER     0x1900020   /* Address of transmit channel enable.  */
#define McBSP1_PCR      0x1900024   /* Address of pin control reg.          */

/* Define L2 Cache Registers */
#define L2CFG           0x1840000   /* Address of L2 config reg             */
#define MAR0            0x1848200   /* Address of mem attribute reg         */

/* Define Interrupt Registers */
#define IMH             0x19c0000   /* Address of Interrupt Multiplexer High*/
#define IML             0x19c0004   /* Address of Interrupt Multiplexer Low */

/* Define Timer0 Registers */
#define TIMER0_CTRL     0x1940000   /* Address of timer0 control reg.       */
#define TIMER0_PRD      0x1940004   /* Address of timer0 period reg.        */
#define TIMER0_COUNT    0x1940008   /* Address of timer0 counter reg.       */

/* Define Timer1 Registers */
#define TIMER1_CTRL     0x1980000   /* Address of timer1 control reg.       */
#define TIMER1_PRD      0x1980004   /* Address of timer1 period reg.        */
#define TIMER1_COUNT    0x1980008   /* Address of timer1 counter reg.       */

/* Define EDMA Registers */
#define PQSR        0x01A0FFE0  /* Address of priority queue status     */
#define CIPR        0x01A0FFE4  /* Address of channel interrupt pending */
#define CIER        0x01A0FFE8  /* Address of channel interrupt enable  */
#define CCER        0x01A0FFEC  /* Address of channel chain enable      */
#define ER        0x01A0FFF0  /* Address of event register            */
#define EER        0x01A0FFF4  /* Address of event enable register     */
#define ECR        0x01A0FFF8  /* Address of event clear register      */
#define ESR        0x01A0FFFC  /* Address of event set register        */

/* Define EDMA Transfer Parameter Entry Fields */
#define OPT        0*4        /* Options Parameter                    */
#define SRC        1*4        /* SRC Address Parameter                */
#define CNT        2*4        /* Count Parameter                      */
#define DST        3*4        /* DST Address Parameter                */
#define IDX        4*4        /* IDX Parameter                        */
#define LNK        5*4        /* LNK Parameter                        */

/* Define EDMA Parameter RAM Addresses */
#define EVENT0_PARAMS 0x01A00000
#define EVENT1_PARAMS EVENT0_PARAMS + 0x18
#define EVENT2_PARAMS EVENT1_PARAMS + 0x18
#define EVENT3_PARAMS EVENT2_PARAMS + 0x18
#define EVENT4_PARAMS EVENT3_PARAMS + 0x18
#define EVENT5_PARAMS EVENT4_PARAMS + 0x18
#define EVENT6_PARAMS EVENT5_PARAMS + 0x18
#define EVENT7_PARAMS EVENT6_PARAMS + 0x18
#define EVENT8_PARAMS EVENT7_PARAMS + 0x18
#define EVENT9_PARAMS EVENT8_PARAMS + 0x18
#define EVENTA_PARAMS EVENT9_PARAMS + 0x18
#define EVENTB_PARAMS EVENTA_PARAMS + 0x18
#define EVENTC_PARAMS EVENTB_PARAMS + 0x18
#define EVENTD_PARAMS EVENTC_PARAMS + 0x18
#define EVENTE_PARAMS EVENTD_PARAMS + 0x18
#define EVENTF_PARAMS EVENTE_PARAMS + 0x18
#define EVENTN_PARAMS EVENTF_PARAMS + 0x18
#define EVENTO_PARAMS EVENTN_PARAMS + 0x18

/* Define QDMA Memory Mapped Registers */
#define QDMA_OPT        0x02000000    /* Address of QDMA options register     */
#define QDMA_SRC        0x02000004    /* Address of QDMA SRC address register */
#define QDMA_CNT        0x02000008    /* Address of QDMA counts register      */
#define QDMA_DST        0x0200000C    /* Address of QDMA DST address register */
#define QDMA_IDX        0x02000010    /* Address of QDMA index register       */

/* Define QDMA Pseudo Registers */
#define QDMA_S_OPT        0x02000020    /* Address of QDMA options register     */
#define QDMA_S_SRC        0x02000024    /* Address of QDMA SRC address register */
#define QDMA_S_CNT        0x02000028    /* Address of QDMA counts register      */
#define QDMA_S_DST        0x0200002C    /* Address of QDMA DST address register */
#define QDMA_S_IDX        0x02000030    /* Address of QDMA index register       */



/* EMIF Parameters
```

```
  Bit positions for C6211 CE Space control registers  */
#define RHOLD        0                 /* read hold */
#define WHOLDMSB    3            /* write hold MSB */
#define MTYPE         4                  /* memory Type */
#define RSTROBE        8            /* read strobe */
#define RSET        16            /* read setup */
#define WHOLD        20            /* Write Hold */
#define WSTROBE        22            /* write stobe */
#define WSET        28            /* write setup */
#define TATIME        14            /* Turn around time */

#define MTYPE_A16    0x1
#define MTYPE_A32    0x2
#define MTYPE_SDRAM8    0x8
#define MTYPE_SDRAM32    0x3
#define MTYPE_SDRAM16    0x9

```

DSP DMA Resources

The file `c6211dma.h` contains the bit position of DMA transfer options and a list of already prepared options:

```
/* definitions for DMA option register */

#define FS_B        0
#define TCC_B        16
#define TCINT_B        20
#define DUM_B        21
#define DD_B        23
#define SUM_B        24
#define DS_B        26
#define ESIZE_B        27
#define PRI_B        29

#define FS        1
#define TCC(bit)     (bit << TCC_B)
#define TCINT        (1 << TCINT_B)
#define DUM_NOMOD    (0 << DUM_B)
#define DUM_INC     (1 << DUM_B )
#define DUM_DEC        (2 << DUM_B)
#define DUM_INDX     (3 << DUM_B)
#define SUM_NOMOD    (0 << SUM_B)
#define SUM_INC     (1 << SUM_B )
#define SUM_DEC        (2 << SUM_B)
#define SUM_INDX     (3 << SUM_B)
#define DD        (1 << DD_B)
#define DS        (1 << DS_B)
#define SIZE_B        (2 << ESIZE_B)
#define SIZE_H        (1 << ESIZE_B)
#define SIZE_W        (0 << ESIZE_B)
#define PRI_H        (1 << PRI_B)
#define PRI_L        (2 << PRI_B)

```

SSM Resources

Hardware addresses of the SSM module are included in the file `scanres.h`. To access information in the hit-buffer, the word count of the hit-buffer content and the base address of the buffer are needed. The SSM event number is split into two 16-bit words.

```
//
//
//    defines vme ressources of the SCANNER
//    Marc R.Stockmeier
//    created  03.09.1999
//    modified 14.09.1999

#define SCAN_DL350BASE   0x0000 /*direct access of pre-selected DL350 wire*/
#define SCAN_CSR         0x4000 /*control&status-register CSR*/
```

```
#define SCAN_SERNUM      0x4001 /*serial number of SSM (read only)*/
#define SCAN_THRES       0x4002 /*threshold-register*/
#define SCAN_RANGE       0x4003 /*scanrange-register*/
#define SCAN_OFFSET      0x4004 /*scanoffset-register*/
#define SCAN_SWIRE       0x4005 /*startwire-register*/
#define SCAN_EWIRE       0x4006 /*endwire-register*/
#define SCAN_PRES        0x4007 /*number of presamples*/
#define SCAN_POSTS       0x4008 /*number of postsamples*/
#define SCAN_TIMER       0x4009 /*programmable clock-generator ICD2051 serial interface*/
#define SCAN_EPROM       0x400a /*configuration EEPROM AT17C256 serial interface*/
#define SCAN_SPARE       0x400b /*8 spare I/Os for future enhancements*/
#define SCAN_SPAREOE     0x400c /*output enable for spare I/Os*/
#define SCAN_WIREREG     0x400d /*DL350-wire-register (selects the FADC wire to access)*/

#define SCAN_STOPADD     0x4010 /*FADC-address where the sampling stopped (read only)*/
#define SCAN_WCOUNT      0x4011 /*word-count (number of words in hit-memory) (read only)*/
#define SCAN_LOWEVNTNUM  0x4012 /*event number (low word) (read only)*/
#define SCAN_HIGHEVNTNUM 0x4013 /*event number (high word) (read only)*/
#define SCAN_STARTSAMP   0x4020 /*start sampling (write only) strobe reg*/
#define SCAN_STOPSAMP    0x4021 /*stop sampling (write only)*/
#define SCAN_STARTSCAN   0x4022 /*start scanning (write only)*/
#define SCAN_NIMPULS     0x4023 /*generate pulse at the NIM-pulse-output*/
#define SCAN_RESEVTCNT   0x4024 /*clear event counter (write only)*/

#define SCAN_HBUFF       0x8000 /*access to 32Kx16 hit buffer memory*/

/*control and status register bits*/
#define CSR_AUTOSMP      0x0001 /*auto sampling start mode*/
#define CSR_AUTOSCAN     0x0002 /*auto scanning start mode*/
#define CSR_EXTCYC       0x0004 /*extended DL350 cycle*/
#define CSR_ENABLENIM    0x0008 /*enable NIM frontpanel signals*/
#define CSR_ENABLEGTB    0x0010 /*enable data output to GTB during scanning*/
#define CSR_PDC4         0x0040 /*position dependent code bit 4 (left/right side of crate) READONLY*/
#define CSR_SAMPLING     0x0080 /*sampling in progress READONLY*/
#define CSR_FPGAVER      0xff00 /*fpga version READONLY*/

/* timer chip 2051 serial interface lines for programming*/
#define TIMER_SCLKA        0x01 /*serial clock A*/
#define TIMER_SCLKB        0x02 /*serial clock B*/
#define TIMER_DATA         0x04 /*serial data*/
#define TIMER_MUXREFA_     0x08 /*multiplex clock A*/

```

SAM3 DSP Resources

The corresponding hardware addresses of the SAM3 module which are needed by the DSP are listed in the file `sam3c60res.h`. Together with the base addresses of the four external memory spaces (CE0-CE3), the starting addresses of GTBRAM and VRAM are defined (lines 5-14). The registers for the communication with the SSM are described in lines 21–25.

```
 1 // SAM3 C60 resources
 2
 3 // chip select
 4
 5 #define DSP_CE0 0x80000000
 6 #define DSP_CE1 0x90000000
 7 #define DSP_CE2 0xa0000000
 8 #define DSP_CE3 0xb0000000
 9
10 // addresses of resources
11
12 #define DSP_ERS     (DSP_CE0 + (0x50001<<2))     // FPGAS DSPLOG and GTB1
13 #define DSP_VMERAM     (DSP_CE1 + (0x10000<<2))     // VMERAM
14 #define DSP_GTBRAM     (DSP_CE1 + (0x20000<<2))     // GTBRAM
15 #define DSP_SDRAM    DSP_CE3
16
17 #define DSP_DISP      (DSP_CE0 + (0x60000<<2))     // ??
18 #define DSP_DISP1      (DSP_CE0 + (0x60001<<2))
19 #define DSP_DISP2      (DSP_CE0 + (0x60002<<2))
20
21 #define DSP_GTBXMOD    (DSP_CE2 + (0x8000d<<2))     // GTB mode (reset(0),address(1),token(3))
22 #define DSP_GTBXADR    (DSP_CE2 + (0x80002<<2))     // GTB address register (lower 16 bit)
23 #define DSP_GTBXDAT    (DSP_CE2 + (0x80000<<2))     // GTB data register (lower 16 bit)
24 #define DSP_GTBXADRPTR    (DSP_CE2 + (0x80004<<2)) // GTBram address pointer counter
25 #define DSP_GTBXDSR     (DSP_CE2 + (0x80008<<2)) // GTB control status register (ACK readout enable,
READY transfer ready)
26
```

```
// host port interface definitions

#define HPI_CTRL_L 0x0
#define HPI_CTRL_H 0x1

#define HPI_ADR_L  0x2
#define HPI_ADR_H  0x3

#define HPI_DATI_L 0x4
#define HPI_DATI_H 0x5

#define HPI_DAT_L  0x6
#define HPI_DAT_H  0x7

// DSR bits

#define DSR_ACK  0x1
#define DSR_RDY  0x2

```

SAM3 VME Resources

Hardware addresses of the SAM3 module which are used for accesses from the VME side (RIO3) are set up in the file `sam3vmeres.h`. The DSP also needs to know about those addresses because it organizes the partitioning of the memories (see below) and has to provide the proper start addresses of the memory blocks for the RIO3.

```

//
//
//    defines vme ressources of SAM with C40 DSP
//    Marc R.Stockmeier
//    created  08.02.1999
//    modified 17.02.1999  *_LEN   added  Marc
//    modified 18.02.1999  *_BASE* added  Marc
//    modified 22.02.1999  *_DISR-L* added Christian
//

#define SAM3_VRAM1       0x00000000  /* VME DPRAM DSP 1 */
#define SAM3_VRAM1_LEN   0x0003FFFC
#define SAM3_VRAM2       0x00100000  /* VME DPRAM DSP 2 */
#define SAM3_VRAM2_LEN   0x0003FFFC

#define SAM3_HPI1        0x00200000  /* Host Port Interface DSP 1 */
#define SAM3_HPI2        0x00300000  /* Host Port Interface DSP 2 */

#define SAM3_CSR0        0x00400000  /* Control Status Register 1 */
#define SAM3_CSR1        0x00400004  /* Control Status Register 2 */

#define SAM3_JTAG       0x00180000   /* JTAG test */
#define SAM3_JTAG_LEN   0x00000000

#define SAM3_FLASH      0x00500000   /* Configuration test */
#define SAM3_FLASH_LEN  0x0000FFFC


//CSR0 bit definitions

#define CSR0_LED1        0x0001  /* LED1 active high */
#define CSR0_LED2        0x0002  /* LED2 active high */
#define CSR0_DSP1_RES    0x0004  /* DSP 1 reset if Low */
#define CSR0_DSP2_RES    0x0008  /* DSP 2 reset if Low */
#define CSR0_DSP1_HOLD   0x0010  /* DSP 1 hold if low */
#define CSR0_DSP2_HOLD   0x0020  /* DSP 2 hold if low */

//CSR1 bit definitions

#define CSR1_FLASH       0x0001  /* FLASH memory */
#define CSR1_VMELOG      0x0008  /* FPGA vmelog  */
#define CSR1_DSPLOG      0x0010  /* FPGA dsplog  */
#define CSR1_GTBL1       0x0020  /* FPGA gtbl1   */
#define CSR1_GTBL2       0x0040  /* FPGA GTBL2   */

```

Memory Partitioning and Error Codes

The partitioning of the GTBRAM and the VRAM allows to buffer incoming events before and after the data reduction step. Thus, the dead time of the DAQ system is reduced. All parameters and addresses needed for the partitioning are defined in the file `mem_part.h`. Furthermore, a small part of the VRAM is configured for the communication between DSP and RIO3 and for keeping information about the contents of the memory blocks in the VRAM. In addition, error codes are defined. Note that before including this file into a (sub)program, the files `sam3c60res.h` and `sam3vmeres.h` have to be included.

At the beginning, the size of one parameter (RLEN, in bytes), the number of DSP program options (NGLP), the number of memory blocks (NBLKS) and the number of parameters which describe the content of a block (NOPTS) have to be set (lines 15–21). Afterwards, the lengths of GTBRAM and VRAM are calculated in units of bytes, shorts (2 B), words (4 B) and 'quads' (8 B) (lines 31–53). The the block sizes are determined and the start addresses of the first blocks are calculated (lines 69–81). Base pointers to the communication memory block in the VRAM and to the header of the first memory block are set up (lines 91–130) and finally, status words and error codes for the communication between DSP and RIO3 are defined (lines 142–153).

```
 1 //
 2 // GTB & VME memory partitioning for DSP & RIO
 3 //   Markus Merschmeyer, 25.04.2002
 4 //
 5 //  caution: ’sam3c60res.h’ & ’sam3vmeres.h’ must be included before !!!
 6 //
 7 //  |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
 8 //       10        20        30        40        50        60        70        80        90        100
 9
10
11 /*****************/
12 /* global options */
13 /*****************/
14
15 #define RLEN           4                                   // length of "registers" (4B)
16 #define NGLP           4                                   // # of global parameters
17 #define NBLKS          2                                   // # of param./memory blocks in GTBRAM...
18                                                            // ...and VRAM: only powers of 2!!!
19 #define NOPTS          5                                   // # of options per block
20 #define BOPT_OFFS      (RLEN*NOPTS)                       // offset for data block options
21 #define OPT_LEN        (((NGLP+NBLKS*NOPTS)/2)+1)*2*RLEN // length of option field (quad aligned)
22
23
24 /***********************************/
25 /* GTBRAM size & partitioning (DSP) */
26 /***********************************/
27 //
28 //  ->  64k x 16bit = 128kB (from SSM/DSP)
29 //  base address is ’DSP_GTBRAM’ (-> ’sam3c60res.h’)
30 //
31 #define GTBRAM_LEN_B  0x00020000                          // length in ’byte’  (1B) (128k)
32 #define GTBRAM_LEN_S  (GTBRAM_LEN_B>>1)                    // length in ’short’ (2^1B = 2B)
33 #define GTBRAM_LEN_W  (GTBRAM_LEN_B>>2)                    // length in ’word’  (2^2B = 4B)
34
35
36 /*************************/
37 /* VMERAM size (DSP & RIO) */
38 /*************************/
39 //
40 //  ->  32k x 32bit = 128kB (from DSP/RIO)
41 //  ->  16k x 64bit = 128kB (from RIO)
```

```
//  DSP : base address is 'DSP_VMERAM' (-> 'sam3c60res.h')
//  RIO : only offsets (relative to VRAM1 on SAM3) are given
//
#define VMERAM_LEN_B  0x00020000                        // (DSP) length in 'byte'  (1B) (128k)
#define VMERAM_LEN_S  (VMERAM_LEN_B>>1)                  // (DSP) length in 'short' (2^1B = 2B)
#define VMERAM_LEN_W  (VMERAM_LEN_B>>2)                  // (DSP) length in 'word'  (2^2B = 4B)
#define VMERAM_LEN_Q  (VMERAM_LEN_B>>3)                  // (DSP) length in 'quad'  (2^3B = 8B)

#define VRAM_LEN_B    0x00020000                        // (RIO) length in 'byte'  (1B) (128k)
#define VRAM_LEN_S    (VRAM_LEN_B>>1)                    // (RIO) length in 'short' (2^1B = 2B)
#define VRAM_LEN_W    (VRAM_LEN_B>>2)                    // (RIO) length in 'word'  (2^2B = 4B)
#define VRAM_LEN_Q    (VRAM_LEN_B>>3)                    // (RIO) length in 'quad'  (2^3B = 8B)


/**********************/
/* data blocks (sizes) */
/**********************/

#define GBLK_SIZE     (GTBRAM_LEN_B>>(NBLKS/2))          // size of block in GTBRAM (128kB/NBLKS)
#define DBLK_SIZE     (VMERAM_LEN_B>>(NBLKS/2))          // size of block in VMERAM (128kB/NBLKS)
#define RBLK_SIZE     (VRAM_LEN_B>>(NBLKS/2))            // size of block in VRAM (128kB/NBLKS)


/****************************************/
/* data blocks (start addresses/offsets) */
/****************************************/
// GTBRAM
#define GTB_DATA_0    (DSP_GTBRAM + GBLK_SIZE*0)       // start addr. of 0th block
//
// next blocks: GTB_DATA_x = GTB_DATA_0 + x*GBLK_SIZE
//

// VMERAM (DSP)
#define DSP_VME_D0    (DSP_VMERAM + DBLK_SIZE*0 + OPT_LEN)  // st. adr. 0th block (DSP) + options!
//
// next blocks: DSP_VME_Dx = DSP_VME_D0 + x*DBLK_SIZE - OPT_LEN
//

// VRAM (RIO) (only offsets to SAM3_VRAM1 !!!)
#define RIO_VR1_D0    (RBLK_SIZE*0 + OPT_LEN)          // offs. for 0th block (RIO) + options!
//
// next blocks: RIO_VR1_Dx = RIO_VR1_D0 + x*RBLK_SIZE - OPT_LEN
//

/***************************/
/* VMERAM partitioning (DSP) */
/***************************/

// addresses of global option/parameter "registers"
#define DSP_OPT_NUM   (DSP_VMERAM + RLEN*0)            // DSP number (1 / 2)
#define DSP_OPT_CTL   (DSP_VMERAM + RLEN*1)            // DSP<->RIO communication "register"
#define DSP_OPT_CP0   (DSP_VMERAM + RLEN*2)            // crate number / programm mode...
                                                        // ...(reduce/dump/both/...)...
                                                        // ...in upper 16bit / lower 16bit
#define DSP_OPT_PPS   (DSP_VMERAM + RLEN*3)            // number of pre- / postsamples...
                                                        // ...in upper 16bit / lower 16bit
// addresses of block-specific option/parameter "registers"
#define DSP_OPT_ST0   (DSP_VMERAM + RLEN*4)            // block 0: status for data in this block
#define DSP_OPT_DA0   (DSP_VMERAM + RLEN*5)            // block 0: start address for data
#define DSP_OPT_DL0   (DSP_VMERAM + RLEN*6)            // block 0: length of data (in bytes)
#define DSP_OPT_DE0   (DSP_VMERAM + RLEN*7)            // block 0: DSP evt. # for data
#define DSP_OPT_SE0   (DSP_VMERAM + RLEN*8)            // block 0: SSM evt. # for data
//
// following blocks: DSP_OPT_yyz = DSP_OPT_yy0
//                                + z*BOPT_OFFS         // parameter block size offset


/***************************/
/* VMERAM partitioning (RIO) */
/***************************/
//
// only offsets to SAM3_VRAM1 !!!
//
#define VRAM_OFFS     (SAM3_VRAM2 - SAM3_VRAM1)        // offset VR1 -> VR2 for RIO

// addresses of global option/parameter "registers" for DSP1
#define RIO_OP1_NUM   (RLEN*0)                          // DSP number (1 / 2)
#define RIO_OP1_CTL   (RLEN*1)                          // DSP<->RIO communication "register"
#define RIO_OP1_CP0   (RLEN*2)                          // crate number / programm mode...
                                                         // ...(reduce/dump/both/...)...
                                                         // ...in upper 16bit / lower 16bit
#define RIO_OP1_PPS   (RLEN*3)                          // number of pre- / postsamples...
                                                         // ...in upper 16bit / lower 16bit
// addresses of block-specific option/parameter "registers"
#define RIO_OP1_ST0   (RLEN*4)                          // block 0: status for data in this block
#define RIO_OP1_DA0   (RLEN*5)                          // block 0: start address for data
#define RIO_OP1_DL0   (RLEN*6)                          // block 0: length of data (in bytes)
#define RIO_OP1_DE0   (RLEN*7)                          // block 0: DSP evt. # for data
#define RIO_OP1_SE0   (RLEN*8)                          // block 0: SSM evt. # for data
```

```
//
// following blocks: RIO_OPx_yyz = RIO_OP1_yy0
//                                 + (x-1)*VRAM_OFFS     // 2nd DSP (VRAM size) offset
//                                 + z*BOPT_OFFS         // parameter block size offset



/********************************************************/
/* DSP<->RIO global communication "register" values (!) */
/********************************************************/

#define COM_RegReset       0x00000000                    // comm. "register" reset
#define COM_ProgStart      0x00000001                    // (DSP) DSP prg. started
#define COM_SSM            0x00000002                    // (DSP) SSM (GTB) initialized
#define COM_MainLoop       0x00000003                    // (DSP) running main loop
#define COM_WaitForAck     0x00000004                    // (DSP) wait f. SSM to react to 'Ack'
#define COM_StartToCopy    0x00000005                    // (DSP) start copying data
#define COM_StartAnalysis  0x00000006                    // (DSP) start data reduction
#define COM_DataRdy        0x00000007                    // (DSP) data in VRAM
#define COM_DataRetrieved  0x00000008                    // (RIO) data retrieved

#define COM_Error          0x00FF00FF                    // error occured
#define COM_Error_BadSLen  0x01FF01FF                    // bad signal length in 'analys'
#define COM_Error_BadNPk   0x02FF02FF                    // bad # of peaks in 'sigan'
#define COM_Error_BigEvent 0x0bFF0bFF                    // event too big
#define COM_Error_BadEvNo  0x0cFF0cFF                    // DSP & SSM evt. #s dont match
#define COM_Error_EvNoWE   0x0dFF0dFF                    // DSP/SSM evt. # write error
#define COM_Error_BadEvent 0x0eFF0eFF                    // bad event ( no 2*0xffff )



/***************************/
/* program mode definitions */
/***************************/

#define MOD_RAW    (1<<0)                                  // 'raw' data mode bit
#define MOD_PTR    (1<<1)                                  // 'pushthrough' mode bit
#define MOD_RED    (1<<2)                                  // 'reduce' mode bit
#define MOD_CH1    (1<<3)                                  // 'compare HB & GTBRAM' mode bit
#define MOD_CH2    (1<<4)                                  // 'check QDMA GR->IR' mode bit
#define MOD_CH3    (1<<5)                                  // 'check QDMA IR->VR' mode bit
#define MOD_CH4    (1<<6)                                  // 'GTBRAM test' mode bit
#define MOD_CH5    (1<<7)                                  // 'VRAM test' mode bit


/************************/
/* DSP internal RAM size */
/************************/
//
//  -> IRAM size: 64kB (for c6x11 DSP types)
//
#define IRAM_LEN_B    0x10000                            // DSP int. RAM length in bytes
#define DSP_IRAM      0x0000                             // start addr. of DSP int. RAM

#define IBLK_SIZE     0x8000                             // length of data block in IRAM (32k)
#define DSP_IRAM_D    (DSP_IRAM+(IRAM_LEN_B-IBLK_SIZE)) // start addr. of data block

#define IBLK_SIZ2     0xC000                             // length of data block in IRAM (48k)
#define DSP_IRAM_D2   (DSP_IRAM+(IRAM_LEN_B-IBLK_SIZ2)) // start addr. of data block

```

DSP Program Parameters

All program parameters are given in the file `redu_defs.h`. First, values for the nominal, the minimum and the maximum allowed FADC pedestal are set. They are needed for the case that the pedestals determined from pre- and post-samples lie outside the allowed range (lines 1–3). Parameters for the peak finding and integration are given in lines 5–8. DPLIM is the threshold for a valid slope at the start of a pulse. ITLIM and IQLIM are the minimum number of valid samples in the rising part of the pulse and the maximum number of samples to be integrated after the pulse's maximum, respectively. HITLENGTH defined the minimum number of samples of a valid pulse. Lines

18–23 contain FADC parameters for the linearization correction, NL is the FADC compression factor multiplied by 100.

Dimensions of arrays needed for linearization and peak search (see below in `sigan.c`) are defined in lines 34–50. Depending on the type of the reduction program (48 k version, fast (final) version or other) different dimensions can be selected by defining either 'SHORT_ARRAYS', 'LONG_ARRAYS' or nothing for the compiler.

```
 1 #define NOM_PEDESTAL 10                        // nominal FADC pedestal
 2 #define MIN_PEDESTAL NOM_PEDESTAL-3            // min. FADC pedestal
 3 #define MAX_PEDESTAL NOM_PEDESTAL+3            // max. FADC pedestal
 4
 5 #define DPLIM        4                         // threshold for valid dp bins
 6 #define ITLIM        2                         // min. # of valid dp bins
 7 #define IQLIM        7                         // min. # of valid Q bins
 8 #define HITLENGTH    8                         // min. # of valid samples/bins for hit
 9
10 //#define PRESAMPLE  5                            // fixed # of presamples
11 //#define POSTSAMPLE 0                            // fixed # of postsamples
12
13 #define PRES_OFFS    2                         // # presamples to include in integration;
14                                                // hopefully no longer needed when SSM works
15
16 //#define NL_MIN       64                         // lowest possible lin. parameter
17 //#define NL_MAX       73                         // highest possible lin. parameter
18 #define NL_MIN       68                        // -> use only one parameter
19 #define NL_MAX       68                        // ...(23.08.2002)...
20 #define NL           68                        // FACD linearization parameter
21 #define MPH          256                       // max. pulse height from FADC
22 #define MAXWIRES     108                       // max. # wires per crate
23 #define N_PAR_WIRE   4                         // # parameters per wire
24
25
26 //#define OVFLOWN       253                       // non-lin. overflow value
27 //#define OVFL         (780-1)                    // linearised overflow value
28 //#define OVFL         (790-1)                    // linearised overflow value (lpar=68)
29 #define OVFLRISE     100                       // max. allowed pulse slope
30
31
32 //// Array dimensions
33 // ...for reduce_tm_48k...
34 #ifdef SHORT_ARRAYS
35   #define ADIM1        500                     // array dimension (p,dp)
36   #define ADIM2        40                      // array dimension (Start,Max,...)
37 #endif
38 // ...for reduce_tm_blk_fast...
39 #ifdef LONG_ARRAYS
40   #define ADIM1        620                     // array dimension (p,dp)
41   #define ADIM2        50                      // array dimension (Start,Max,...)
42 #endif
43
44 // ...for any other reduce_*...
45 #ifndef SHORT_ARRAYS
46   #ifndef LONG_ARRAYS
47     #define ADIM1        512                   // array dimension (p,dp)
48     #define ADIM2        50                    // array dimension (Start,Max,...)
49   #endif
50 #endif
51
52
53
54 // define only if rise time of pulses (l/r)...
55 //  ...shall also be saved !!!
56 ////#define RISETIME
57
58
59 /*** some additional definitions ***/
60 #define HB_LEN       0x8000                     // SSM hit buffer length (32k x 16bit = 64 kB)
61 #define EN_OFFS      1                          // 32bit offset due to event # at start of SSM data
62 #define SIG_OFFS     2                          // signal offset (wire # & time offs. values)...
63                                                 // ...(-> 'pushthrough.c' & 'analys.c')
```

Global Variables

Global variables for all reduction programs are declared in `redu_glovar.h`. In lines 4-7 the lengths of the types 'short' and 'int' are determined and the FADC overflow threshold is set. Note that this is just a preset value. The actual threshold is calculated with respect to the FADC compression parameter in the subprogram `calc_lin_corr.c`.

Global pointers to the raw data and to the arrays for the linearization correction are defined in lines 13–28, variables for the pedestal correction are declared in lines 33–36. Arrays used for correction of the integration cutoff and of the background of a preceding pulse are set up in lines 44–50 and the arrays for the data reduction are defined in lines 72–83. For all subprograms, the include file `redu_glovar_ext.h` has to be used.

```
 1
 2 // global type sizes
 3
 4 const unsigned int L_SHORT = sizeof(short); // size of 'short' in bytes
 5 const unsigned int L_INT   = sizeof(int);   // size of 'int' in bytes
 6
 7       unsigned int OVFL    = (780-1);       // overflow threshold
 8
 9 // global pointers
10
11 //volatile unsigned short *psig;              // pointer to raw data (signals)
12 //      unsigned short *psig;              // pointer to raw data (signals)
13 const unsigned short *psig;              // pointer to raw data (signals)
14
15 /*
16 volatile unsigned short *pres;              // pointer to reduced data (results)
17 volatile unsigned int   *ptr_res;           // 'int' pointer to reduced data (results)
18 volatile unsigned short *linbase;           // pointer to linearisation value array
19 volatile unsigned short *linparm;           // pointer to linearisation param. array
20 volatile unsigned short *linpl;             // pointer f. linearisation correction, left
21 volatile unsigned short *linpr;             // pointer f. linearisation correction, right
22 */
23       unsigned short *pres;              // pointer to reduced data (results)
24       unsigned int   *ptr_res;           // 'int' pointer to reduced data (results)
25       unsigned short *linbase;           // pointer to linearisation value array
26       unsigned short *linparm;           // pointer to linearisation param. array
27       unsigned short *linpl;             // pointer f. linearisation correction, left
28       unsigned short *linpr;             // pointer f. linearisation correction, right
29
30
31 // global values/variables
32
33       unsigned int   pre_smpl;           // number of presamples
34       unsigned int   pst_smpl;           // number of postsamples
35       unsigned int   bgrdl;              // backgr./baseline from pre-/postsamples, left
36       unsigned int   bgrdr;              // backgr./baseline from pre-/postsamples, right
37
38
39 // empiric constants for integration-cutoff correction
40
41 //const unsigned int cor[16] = {267,229,206,188,175,164,156,150,
42 //                              145,142,139,136,133,132,131,129};
43
44 const unsigned int cor[8] = {229,188,164,150,142,136,132,129};
45
46
47 // empiric constants for pre-pulse background correction
48
49 const unsigned int bgc[16] = { 67, 56, 49, 41, 35, 28, 23, 19,
50                                15, 13, 10,  8,  6,  4,  2,  1};
51
52
53 // MM special variables for signal analysis
54 /*
55   volatile unsigned   int p_arr[ADIM1];     // linearised pulse sample values (l/r)
56   volatile          short dp_arr[ADIM1];    // comb. differential pulse samples
57   volatile unsigned   int sig_map[ADIM2];   // signal marks: |2*0| |10*a| |10*b| |10*c|
58                                             //  a: signal-start index
59                                             //  b: signal-maximum index
60                                             //  c: signal-end index
61   volatile            int time_val[ADIM2];  // time integration results: |16*left| |16*right|
#ifdef RISETIME
  volatile            int dpm_val[ADIM2];   // mean rise time: |16*left| |16*right|
#endif
  volatile            int q_val[ADIM2];     // charge integration results: |16*left| |16*right|
  volatile unsigned   int oftq_val[ADIM2];  // important indices: # overflow bins,
                                            //                      # q/t integration bins
                                            // |8*ovfl. l.| |8*ovfl. r.| ...
                                            // ... |8*i_charge (iq)| |8*i_time (it)|
*/

  unsigned   int p_arr[ADIM1];     // linearised pulse sample values (l/r)
           short dp_arr[ADIM1];    // comb. differential pulse samples
  unsigned   int sig_map[ADIM2];   // signal marks: |2*0| |10*a| |10*b| |10*c|
                                   //  a: signal-start index
                                   //  b: signal-maximum index
                                   //  c: signal-end index
             int time_val[ADIM2];  // time integration results: |16*left| |16*right|
#ifdef RISETIME
             int dpm_val[ADIM2];   // mean rise time: |16*left| |16*right|
#endif
             int q_val[ADIM2];     // charge integration results: |16*left| |16*right|
  unsigned   int oftq_val[ADIM2];  // important indices: # overflow bins,
                                   //                      # q/t integration bins
                                   // |8*ovfl. l.| |8*ovfl. r.| ...
                                   // ... |8*i_charge (iq)| |8*i_time (it)|

```

D.3.3 Setup Programs

The setup programs are needed for the preparation of the external memory interface and the linearization correction lookup tables.

Memory Configuration

The program `sam3_memconfig.c` defines the timing parameters for the four memory address spaces (CE0–CE3) of the DSP (lines 99–134). The GT-BRAM and the VRAM address spaces lie within CE1, the GTB control registers are mapped into CE2.

```
 1 //
 2 // original by W.Ott
 3 // modified by M.Merschmeyer, 18.05.2002
 4 //
 5
 6 #include "c6211.h"                               // c6x11 global definitions
 7
 8 // CE0 :  LED + registers
 9 // CE1 : dual ported VME memory & GTB memory
10 // CE2 : GTB registers
11 // CE3 : SDRAM
12
13 //  |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
14 //       10        20        30        40        50        60        70        80        90      100
15
16
17 //
18 // EMIF Registers (c621x/c671x): (page 10-14)
19 //     -> see 'TMS320C6000 Peripherals Reference Guide' for futher details
20 //        (document code: SPRU190D; chapter 10)
21 //
22 //   EMIF_GCR    : global control (pp. 10-15/10-16/10-17)
23 //                 (@ : set default values explicitly)
24 //
25 //      31 - 24 : |                           Rsv                              |
26 //                                             R,+0
27 //      23 - 16 : |                           Rsv                              |
28 //                                             R,+0
29 //      15 -  8 : | Rsv   | Rsv@ | Rsv@ | Rsv@ | BUSREQ | ARDY | HOLD | HOLDA |
30 //                   R,+0   RW,+0  RW,+1  RW,+1  R,+0     R,+x   R,+x   R,+x
31 //       7 -  0 : | NOHOLD | Rsv  | Rsv  | CLK1EN | CLK2EN | Rsv | Rsv | Rsv |
32 //                   RW,+0    R,+1   R,+1  RW,+1    RW,+1    R,+0  R,+0  R,+0
33 //
34 //   EMIF_CE0    : CE0 control (pp. 10-19/10-20)
```

```
//   EMIF_CE1    : CE1 control
//   EMIF_CE2    : CE2 control
//   EMIF_CE3    : CE3 control
//
//      31 - 22 : |         Write setup        |         Write strobe       |
//                           RW,+1111                     RW,+111111
//      21 - 16 : |         Write hold         |          Read setup        |
//                            RW,+11                       RW,+1111
//      15 -  8 : |             TA             |          Read strobe       |
//                             R,+11                       RW,+111111
//       7 -  0 : |        MTYPE       |   Rsv   |        Read hold         |
//                        RW,+0010         R,+0            RW,+011
//
//   EMIF_SDCTRL : SDRAM control (pp. 10-23/10-24)
//
//      31 - 26 : |      Rsv      |     SDBSZ    |     SDRSZ    |    SDCSZ    |
//                        R,+0          RW,+0         RW,+00         RW,+00
//      25 - 16 : |     RFEN      |     INIT     |     TRCD     |     TRP     |
//                     RW,+1           W,+1        RW,+0100       RW,+1000
//      15 -  0 : |           TRC            |            Reserved            |
//                          RW,+1111                R,+0000 0000 0000
//
//   EMIF_SDRP   : SDRAM refresh period (pp. 10-26)
//
//      31 - 24 : |         Reserved         |             XRFR             |
//                         R,+0000 00                    RW,+00
//      23 -  0 : |         COUNTER          |            PERIOD            |
//                     RW,+0101 1101 1100           R,+0101 1101 1100
//

void sam3_memconfig()
{

  *(volatile int *)EMIF_GCR  = 0x00003300;

//
//  memory configuration bit patterns (original)
//
//         WSET WSTROBE WHOLD RSET TATIME RSTROBE MTYPE  RHOLD
// CE0  :  0010 001010  10    0010 10     001010  0010 0 010
// CE1  :  0000 001010  00    0000 10     001111  0010 0 001
// CE2  :  0010 010011  10    0010 10     001111  0010 0 010
// CE3  :  0010 001010  10    0010 10     001010  1000 0 001
//
// (NEW: since 27.08.2002)
// CE1  :  0001 010101  10    0000 10     010000  0010 0 000
// CE2  :  0010 100110  10    0010 10     001111  0010 0 010
//
// (NEW: since 20.08.2002)
// CE1  :  0001 010100  10    0000 10     010000  0010 0 000
//
// (NEW: since 16.07.2002)
// CE1  :  0000 001010  00    0011 10     010100  0010 0 001
//
// (NEW: since 15.07.2002)
// CE1  :  0000 001010  00    0000 10     001111  0010 0 001
// CE2  :  0010 010011  10    0010 10     001111  0010 0 010
//
// (OLD: before 15.07.2002)
// CE1  :  0000 001010  00    0000 10     001100  0010 0 000
// CE2  :  0010 001111  10    0010 10     001111  0010 0 010
//
//

  *(volatile int *)EMIF_CE0  = (2 << RHOLD) +
                               (MTYPE_A32 << MTYPE) +
                               (10 << RSTROBE ) +
                               (2 << RSET) +
                               (2 << WHOLD) +
                               (2 << WSET ) +
                               (10 << WSTROBE) +
                               (2 << TATIME)  ;

  *(volatile int *)EMIF_CE1  = (0 << RHOLD) +
                               (MTYPE_A32 << MTYPE) +
                               (16 << RSTROBE ) +
                               (0 << RSET) +
                               (2 << WHOLD) +
                               (1 << WSET ) +
                               (21 << WSTROBE) +
                               (2 << TATIME)  ;

  *(volatile int *)EMIF_CE2  = (2 << RHOLD) +
                               (MTYPE_A32 << MTYPE) +
                               (15 << RSTROBE ) +
                               (2 << RSET) +
                               (2 << WHOLD) +
                               (2 << WSET ) +
//                               (19 << WSTROBE) +
```

```c
                               (38 << WSTROBE) +
                               (2 << TATIME)  ;

  *(volatile int *)EMIF_CE3  = (1 << RHOLD) +
                               (MTYPE_SDRAM8 << MTYPE) +
                               (10 << RSTROBE ) +
                               (2 << RSET) +
                               (2 << WHOLD) +
                               (2 << WSET ) +
                               (10 << WSTROBE) +
                               (2 << TATIME)  ;
/*
  *(volatile int *)EMIF_SDCTRL = (15 << TRC_B) +
                                 (8 << TRP_B) +
                                 (4 << TRCD_B) +
                                 (1 << INIT) +
                                 (1 << RFEN_B) +
                                 (1 << SDCSZ_B) +
                                 (1 << SDRSZ_B) +
                                 (1 << SDBSZ_B)  ;
*/
  *(volatile int *)EMIF_SDCTRL = 0x57228000;     // 0101 0111 0010 0010 1000 0000 0000 0000

  *(volatile int *)EMIF_SDRP   = 0x0000061a;     // refresh = 1562 EMIF clock cycles

}

```

FADC Compression Factor

The FDAC compression factor (cf. equation 3.1) and a gain correction factor can be set independently for each wire with the program `set_lin_par.c`. At present, the gain correction is done offline and the compression parameter is set to 68 for all wires (FADCs).

```c
void set_lin_par()
//
// set parameters for linearisation
//
{

#include "sam3c60res.h"                                 // SAM3 DSP resources
#include "sam3vmeres.h"                                 // SAM3 VME resources
#include "mem_part.h"                                   // DSP/RIO memory partitioning

#include "redu_defs.h"                                  // special definitions f. data reduction
#include "redu_glovar_ext.h"                            // external (global) variables


//  static unsigned short  p_array[384];                  // linearisation parameter array
  static unsigned short  p_array[MAXWIRES*N_PAR_WIRE];  // linearisation parameter array
         unsigned int    i;                             // loop counter


  linparm = &p_array[0];                                // get start address of array
//  for (i = 0; i < 96; i++) {
  for (i = 0; i < MAXWIRES; i++) {
    *linparm++ = NL;                                    // linearisation parameter (left)
    *linparm++ = NL;                                    // linearisation parameter (right)
    *linparm++ = 128;                                   // preampl. gain corr. factor (left)
    *linparm++ = 128;                                   // preampl. gain corr. factor (right)
  }
  linparm = &p_array[0];                                // save start address of array

}

```

FADC Linearization Correction

A fast linearization of the FADC data is only possible if the correction does not have to be calculated each time a new sample is read. Therefore, the program `calc_lin_corr.c` creates a lookup table which contains the linearized sample values. During the data reduction the correct values can be determined by using the input value as an offset for the base pointer to this array. In addition the overflow threshold for the FADCs is calculated (lines 50–52).

```
 1 void calc_lin_corr()
 2 //
 3 // calculate correction table for FADC linearisation correction
 4 //
 5 {
 6
 7 #include "sam3c60res.h"                               // SAM3 DSP resources
 8 #include "sam3vmeres.h"                               // SAM3 VME resources
 9 #include "mem_part.h"                                 // DSP/RIO memory partitioning
10
11 #include "redu_defs.h"                                // special definitions f. data red.
12 #include "redu_glovar_ext.h"                         // external (global) variables
13
14
15 //  static unsigned short  l_array[2560];               // linearisation value array
16   static unsigned short  l_array[MPH*(NL_MAX-NL_MIN+1)];
17          unsigned short  lpar;                        // parameter index
18          unsigned short  lsig;                        // pulse height index
19
20
21   linbase = &l_array[0];                              // get start address of array
22 //  for (lpar = NL_MIN; lpar < (NL_MAX+1); lpar++) {     // (b) loop over parameter range
23   for (lpar = NL; lpar < (NL+1); lpar++) {             // (b) loop over parameter range
24
25     lsig = 0;                                         // reset while-loop counter
26     while (lsig < MPH) {                               // (b) loop over pulse heights
27       *linbase++ = (25600*lsig)/(25600-lpar*lsig);    // calc. linearised value
28       lsig++;                                         // increase loop counter
29     }                                                 // (e) loop over pulse heights
30
31   }                                                   // (e) loop over parameter range
32
33   linbase = &l_array[0];                              // save start address of array
34
35
36 //(26.11.2003, MM) calc. correct overflow threshold
37 //
38 // The overflow threshold of (790-1) (-> OVFL in redu_defs.h)
39 //  for the linearized pulses was wrong; the FADC writes 0xFF
40 //  (=255) for an overflow, so 0xFE (=254) is the last valid
41 //  pulse height. BUT (!!!) the SSM needs the value 0xFF as a
42 //  mark for the end of a hitlet/pulse so that 0xFE is used as
43 //  the overflow indicator afterwards!!!
44 // => the value 0xFD (=253) is then the NEW last valid pulse
45 //  height value!!!
46 // => to avoid the problem to adjust the overflow threshold
47 //  anytime a new linearization parameter ('lpar') is set,
48 //  from now on the threshold will be calculated online.
49
50   lsig = MPH - 3;                                     // last valid pulse heigth (253)
51   lpar = NL;                                          // FADC linearization parameter
52   OVFL = (25600*lsig)/(25600-lpar*lsig);              // new overflow threshold
53
54
55
56
57 }
```

D.3.4 Data Reduction

The data reduction is performed by the two following programs. In a first step, the raw data are prepared (`analys.c`) and then processed by the reduction algorithm (`sigan.c`).

Signal Preparation

This program loops over the raw data in the internal memory of the DSP. It uses two pointers ('crate', 'ptr_res') to read the raw data and to write the results to the VRAM, respectively. The hitlets in the raw data are separated by a `0xFFFF` marker which is followed by a 16-bit word containing the wire number and another 16-bit word giving FADC sample offset for the calculation of the proper drift time of the pulse. The following data words (up to 5) are the pre-samples and afterwards the samples of the actual pulse. The end of the full event is marked by an extra `0xFFFF` word.

The first step in preparing the data is to determine the length of the actual hitlet which consists of the pre- and post-samples and the pulse itself. If the number of samples in the pulse is above a certain limit, the wire number, sample offset and the pointers to the linearization lookup table are calculated (lines 68–77). Thereafter, the pre-sample background is determined for the pedestal correction (post-samples are not used here) and a pointer to the first sample of the real pulse is created (lines 85–138). Then, the data pointer is set to the first data word of the following hitlet which is either a wire number or another `0xFFFF` marker in case of the end of the event. In addition, a check of the length of the pulse is performed in order ensure that it will fit into the arrays needed for the data reduction (lines 143–152).

The data reduction algorithm is called (line 156) and if it was successful, wire number and data length of the processed hitlet are stored and the pointer for further results is set to the next free data word in the VRAM (lines 159–167). At the end of an event, the size (in bytes) of the resulting data is calculated and an end marker (`0xFFFFFFFF`) is written to the current position in the VRAM. Finally, the byte count of the resulting data is returned to the main program (lines 174–183).

```
 1 unsigned int analys(const unsigned short *crate,
 2                     const unsigned int *rptr)
 3 //
 4 // prepare & start the data reduction algorithm
 5 //
 6 //   crate : pointer to analysis data area
 7 //   rptr  : pointer to results memory area
 8 //
 9 {
10
11 #include "sam3c60res.h"                              // SAM3 DSP resources
12 #include "sam3vmeres.h"                              // SAM3 VME resources
13 #include "mem_part.h"                                // DSP/RIO memory partitioning
14
15 #include "redu_defs.h"                               // special definitions f. data reduction
16 #include "redu_glovar_ext.h"                         // external (global) variables
17
18 // function prototypes
19 unsigned int sigan(const unsigned int, const unsigned int);
20
21 //M1//  volatile unsigned short *crate;                    // pointer to data for analysis
22 //M1           unsigned short *crate;                    // pointer to data for analysis
23            unsigned int    siglen;                   // signal length of hitlet
24            unsigned int    i;                        // loop counter
25
26 //  volatile unsigned short *w;
27            unsigned short *w;
28            unsigned int    bgrdprel;                 // presample backgound, left
```

```
           unsigned int    bgrdprer;                 // presample backgound, right
           unsigned int    bgrdpostl;                // postsample background, left
           unsigned int    bgrdpostr;                // postsample background, right
           unsigned int    wire;                     // current wire number
           unsigned int    roff;                     // time offset for hit
           unsigned int    rlen;                     // length of reduced data
           unsigned int    b_cnt;                    // byte counter for data

           unsigned short  tmp;                      // temporary variable



//M1//  crate  = crate;                                     // set pointer to data address
//M1  crate  = (unsigned short *)crate;                   // set pointer to data address
//  crate += 2;                                        // increase for event #...
//                                                     // ...done in 'get_gtb_data'
  crate += 1;                                        // increase for start marker 0xffff

//  ptr_res = rptr;
  ptr_res = (unsigned int *)rptr;
  b_cnt = 0;                                         // reset byte counter


  while (*crate != 0xFFFF) {                         // (WHILE #1) loop until end of event data

    crate += SIG_OFFS                                // increase for wire # and time offset,...
           + pre_smpl                                // ...for presamples,...
           + pst_smpl                                // ...for postsamples,...
           + ITLIM;                                  // ...and for at least 2 valid signal samples
    siglen = 0                                       // do not increase f. wire # & time offset,...
           + pre_smpl                                // ...for presamples,...
           + pst_smpl                                // ...for postsamples,...
           + ITLIM;                                  // ...and for at least 2 valid signal samples

    while (*crate++ != 0xFFFF) {                     // (WHILE #2) loop until end of hitlet
      siglen++;                                      // ...determine signal length...
    }                                                // (WHILE #2) loop until end of hitlet


    if (siglen > (HITLENGTH+pre_smpl+pst_smpl)) {    // (IF #1) check signal length

      crate -= (siglen+1+SIG_OFFS);                  // set ptr. back to wire #

      wire = _extu((unsigned int)*crate++,24,24);    // get wire number
      roff = 50*_extu((unsigned int)*crate++,22,22); // get time offset

      w     = linparm + N_PAR_WIRE*wire;             // get lin. parameter pointer
      linpl = linbase + MPH*(*(w+0)-NL_MIN);         // base pointer for left lin.
      linpr = linbase + MPH*(*(w+1)-NL_MIN);         // base pointer for right lin.
/*
      gainl = *(w+2);  // gain factor (left)   -> can be corrected offline
      gainr = *(w+3);  // gain factor (right)   => now obsolete
                       // see 'set_lin_par.c'
*/


      if ((pre_smpl-PRES_OFFS) >= 0) {
//M1        psig  = crate + (pre_smpl-PRES_OFFS);        // start integration earlier...
//M2        psig  = (unsigned short *)crate
//M2              + (pre_smpl-PRES_OFFS);                // start integration earlier...
        psig  = crate + (pre_smpl-PRES_OFFS);        // start integration earlier...
      } else {
//M1        psig  = crate + pre_smpl;                    // 'real' start of pulse...
//M2        psig  = (unsigned short *)crate + pre_smpl;  // 'real' start of pulse...
        psig  = crate + pre_smpl;                    // 'real' start of pulse...
      }

      bgrdprel = 0;                                  // reset pres. bgr. sum, left
      bgrdprer = 0;                                  // reset pres. bgr. sum, right
      if (pre_smpl > 0) {                             // determine presample background
//???      for (i=0; i<(pre_smpl-PRES_OFFS); i++) {        // loop over presamples
        for (i=0; i<pre_smpl; i++) {                  // loop over presamples
          bgrdprel += _extu((unsigned int)*crate,16,24);
          bgrdprer += _extu((unsigned int)*crate++,24,24);
        }
        bgrdprel /= pre_smpl;                        // calc. mean background, left
        bgrdprer /= pre_smpl;                        // calc. mean background ,right

        if (bgrdprel < MIN_PEDESTAL) bgrdprel = MIN_PEDESTAL;
        if (bgrdprer < MIN_PEDESTAL) bgrdprer = MIN_PEDESTAL;
        if (bgrdprel > MAX_PEDESTAL) bgrdprel = MAX_PEDESTAL;
        if (bgrdprer > MAX_PEDESTAL) bgrdprer = MAX_PEDESTAL;
      }


////////////////// (tief drueber nachdenken!!!)
////      crate += (siglen-(pre_smpl+pst_smpl));         // set pointer to 1st postsample
//
//      bgrdpostl = 0;                                 // reset posts. bgr. sum, left
//      bgrdpostr = 0;                                 // reset posts. bgr. sum, right
//      if (pst_smpl > 0) {                             // determine postsample background
//        for (i=0; i<pst_smpl; i++) {                  // loop over postsamples
//          bgrdpostl += _extu((unsigned int)*crate,16,24);
//          bgrdpostr += _extu((unsigned int)*crate++,24,24);
//        }
//        bgrdpostl /= pst_smpl;                       // calc. mean background, left
//        bgrdpostr /= pst_smpl;                       // calc. mean background, right
//
//        if (bgrdpostl < MIN_PEDESTAL) bgrdpostl = MIN_PEDESTAL;
//        if (bgrdpostr < MIN_PEDESTAL) bgrdpostr = MIN_PEDESTAL;
//        if (bgrdpostl > MAX_PEDESTAL) bgrdpostl = MAX_PEDESTAL;
//        if (bgrdpostr > MAX_PEDESTAL) bgrdpostr = MAX_PEDESTAL;
//
//        tmp   = (bgrdprel+bgrdpostl)/2;
//        bgrdl = (unsigned int)(*(linpl+tmp));         // calc. linearised mean bgr., left
//        tmp   = (bgrdprer+bgrdpostr)/2;
//        bgrdr = (unsigned int)(*(linpr+tmp));         // calc. linearised mean bgr., right
//      } else {                                        // ...only presample background...
        bgrdl = (unsigned int)(*(linpl+bgrdprel));   // calc. linearised mean bgr., left
        bgrdr = (unsigned int)(*(linpr+bgrdprer));   // calc. linearised mean bgr., right
//      }
////      crate  += 1;                                   //


      siglen -= (pre_smpl + pst_smpl);                 // calc. remaining signal length
      crate  += siglen + pst_smpl + 1;                 // set to next wire number address
/////////////
      siglen += PRES_OFFS;                             // correct for presample offset
/////////////

//////////////////////////////////////////////////////////////////////////////////////////////////
      if (siglen > ADIM1) {                           // signal length too big?
        return COM_Error_BadSLen;                    // return error message
      }
//////////////////////////////////////////////////////////////////////////////////////////////////

      rlen = 0;
      rlen = sigan(siglen,roff);                       // do signal analysis/reduction

//////////////////////////////////////////////////////////////////////////////////////////////////
      if (rlen == COM_Error_BadNPk) {                  // error in 'sigan'?
        return COM_Error_BadNPk;                     // return error message
      }
//////////////////////////////////////////////////////////////////////////////////////////////////

      if (rlen != 0) {
        *ptr_res = ((rlen+L_INT) << 16) + wire;       // write red. data length & wire #
        ptr_res += rlen/L_INT + 1;                   // recalc. results pointer
      }

    }                                                 // (IF #1) check signal length

  }                                                   // (WHILE #1) loop until end of event data


  b_cnt = (unsigned int)ptr_res-(unsigned int)rptr;   // calc. byte counter

  *ptr_res++ = 0xffffffff;                            // write event-end marker

  if (b_cnt > 0) {                                     // data left after reduction...
    b_cnt += L_INT;                                  // ...increase byte counter
  }


  return b_cnt;                                       // return byte counter

}
```

Peak Finding and Data Reduction

The data reduction algorithm is implemented in the program `sigan.c`. At the beginning, all samples of the pulse are linearized an stored in an array for further processing (lines 84–90). The derivative of the combined pulse (both end of the wire) is calculated and the results are written into another array (lines 98–104).

The contents of both arrays are then used in the peak search in order to find start, maximum and end points of peaks in the hitlet and to count possible FADC overflows (lines 143–224).

When the peak search is finished and at least on peak has been detected in the hitlet, its start (drift) time is determined. This is done by calculating the mean position of the rising slope of the peak and by adding the FADC sample offset separately for the signals from both sides of the wire (lines 232–294).

If valid start times could be calculated for the peak, the integral ($\propto$ charge) can be determined for both sides. Summing the sample values from the peak's start up to the pre-set end point after the peak's maximum, yields the uncorrected integral. Subsequently, corrections for contributions by preceding peaks and for losses by the integration cutoff are applied (lines 302–380).

Finally, the results (32 bits for both start times, 32 bits for both integrals and 32 bits for number of overflows, length of the rising slope and length of the integration) are written to the address of the results pointer (lines 388–409).

```
 1 unsigned int sigan(const unsigned int siglen, const unsigned int roff)
 2 {
 3
 4 #include "sam3c60res.h"                // SAM3 DSP resources
 5 #include "sam3vmeres.h"                // SAM3 VME resources
 6 #include "mem_part.h"                  // DSP/RIO memory partitioning
 7
 8 #include "redu_defs.h"                 // special definitions f. data reduction
 9 #include "redu_glovar_ext.h"           // external (global) variables
10
11
12 // MM special variables for signal analysis
13 /*
14   extern volatile unsigned int p_arr[ADIM1];     // linearised pulse sample values (l/r)
15   extern volatile        short dp_arr[ADIM1];   // comb. differential pulse samples
16   extern volatile unsigned int sig_map[ADIM2];  // signal marks: |2*0| |10*a| |10*b| |10*c|
17                                                 //  a: signal-start index
18                                                 //  b: signal-maximum index
19                                                 //  c: signal-end index
20   extern volatile          int time_val[ADIM2]; // time integr.: |16*left| |16*right|
21 #ifdef RISETIME
22   extern volatile          int dpm_val[ADIM2];  // mean rise time: |16*left| |16*right|
23 #endif
24   extern volatile          int q_val[ADIM2];    // charge integr. results: |16*left| |16*right|
25   extern volatile unsigned int oftq_val[ADIM2]; // important indices: # overflow bins,
26                                                 //                    # q/t integration bins
27                                                 // |8*ovfl. l.| |8*ovfl. r.| ...
28                                                 // ... |8*i_charge (iq)| |8*i_time (it)|
29 */
30
31   extern unsigned int p_arr[ADIM1];     // linearised pulse sample values (l/r)
32   extern        short dp_arr[ADIM1];   // comb. differential pulse samples
33   extern unsigned int sig_map[ADIM2];  // signal marks: |2*0| |10*a| |10*b| |10*c|
34                                        //  a: signal-start index
35                                        //  b: signal-maximum index
36                                        //  c: signal-end index
37   extern          int time_val[ADIM2]; // time integr. results: |16*left| |16*right|
38 #ifdef RISETIME
39   extern          int dpm_val[ADIM2];  // mean rise time: |16*left| |16*right|
40 #endif
41   extern          int q_val[ADIM2];    // charge integr. results: |16*left| |16*right|
42   extern unsigned int oftq_val[ADIM2]; // important indices: # ovfl. bins,
43                                        //                    # q/t integration bins
44                                        // |8*ovfl. l| |8*ovfl. r| ...
45                                        // ... |8*i_charge (iq)| |8*i_time (it)|
46
47
48
49   unsigned   int it;                   // time integration bin counter
50   unsigned   int iq;                   // charge integration bin counter
51   unsigned   int ib1,ib2,ib3;          // background calc. indices
52   unsigned   int ql_bg,qr_bg;         // background saving variables
53   unsigned   int iol,ior;              // overflow counter left/right
```

```
  unsigned   int rlen;                // length of red. data (in 32bit words)

             int stq;                 // combined integral value
             int stql,stqr;           // time/charge integral l/r
             int sl,sr;               // pulse height l/r

  unsigned   int sig_smp;             // combined pulse sample value
  unsigned   int p_a,p_b;            // pulse sample values l/r
           short sum_a,sum_b;        // integration variables
             int p_diff;              // pulse difference
  unsigned   int i,j,k;               // general loop counter
  unsigned   int loop_a;              // loop-start point
  unsigned   int loop_b;              // loop-end point
  unsigned   int n_max;               // # of maxima in hitlet




  rlen = 0;                           // reset red. data length
  iol  = 0;                           // reset overflow counters
  ior  = 0;
  ib1  = 0;                           // reset background indices
  ib2  = 0;
  ib3  = 0;


/*********************************************************************/
/*** load pulse samples from memory & do linearisation correction ***/
/*********************************************************************/

#pragma MUST_ITERATE(HITLENGTH+1,ADIM1, );
  for (i=0; i<siglen; i++) {                                    // loop over signal samples
    sig_smp  = (unsigned int)psig[i];                          // load signal sample
    p_a      = (unsigned int)(*(linpl + _extu(sig_smp,16,24))); // linearise pulse sample left
    p_b      = (unsigned int)(*(linpr + _extu(sig_smp,24,24))); // linearise pulse sample right
    p_arr[i] = _extu(p_a,16,0) | p_b;                          // store linearised pulse sample
  }



/****************************************/
/*** calculate ’derivative’ of hitlet ***/
/****************************************/

  sum_a  = (short)((p_arr[0] & 0xffff) + _extu(p_arr[0],0,16)); // calc. 1st pulse sample sum (l/r)
#pragma MUST_ITERATE(HITLENGTH,ADIM1-1, );
  for (i=0; i<siglen-1; i++) {                                  // loop over signal samples
    sum_b = (short)((p_arr[i+1] & 0xffff) + _extu(p_arr[i+1],0,16)); // calc. (i+1)-th sum
    dp_arr[i] = sum_b - sum_a;                                 // calc. difference
    sum_a = sum_b;                                             // save (i+1)-th sum
  }


/* alternative linearisation & differentiation */
/*  sum_a = 0;
#pragma MUST_ITERATE(HITLENGTH+1,ADIM1, );
  for (i=0; i<siglen; i++) {                                    // loop over signal samples
    p_b      = (unsigned int)(*(linpr +  (psig[i] & 0xFF)));
    p_a      = (unsigned int)(*(linpl + ((psig[i]>>8) & 0xFF)));
    p_arr[i] = (p_a << 16) | p_b;

    sum_b     = (short)(p_a + p_b);
    if (i>0) dp_arr[1-i] = sum_b - sum_a;
    sum_a     = sum_b;
  }
*/

/*  sum_a  = 0;
#pragma MUST_ITERATE(HITLENGTH+1,ADIM1, );
  for (i=0; i<siglen; i++) {                                    // loop over signal samples
    sig_smp  = (unsigned int)psig[i];                          // load signal sample
    p_a      = (unsigned int)(*(linpl + _extu(sig_smp,16,24))); // linearise pulse sample left
    p_b      = (unsigned int)(*(linpr + (sig_smp & 0xFF)));     // linearise pulse sample right
    p_arr[i] = (p_a << 16) + p_b;                               // store linearised pulse sample

    sum_b = (short)(p_a + p_b);                                // calc. (i+1)-th sum
    if (i>0) {
      dp_arr[i-1] = sum_b - sum_a;                             // calc. difference
    }
    sum_a = sum_b;                                             // save (i+1)-th sum
  }
*/



/************************************************************/
/* *** analyse pulse/signal shape (=> START / MAX / END) *** */
/************************************************************/

```

```
  n_max = 0;                                                    // reset pulse (maximum) counter
  i = 0;                                                        // reset loop counter
  while (i < siglen) {                                           // (b) loop over signal samples

    if (n_max > 0) {                                             // (b) check for previous pulse

//////////////////////////////////////////////////////////////////////////////////////////////////
      if (n_max >= ADIM2) {                           // too many peaks?
        return COM_Error_BadNPk;                    // return error message
      }
//////////////////////////////////////////////////////////////////////////////////////////////////

      if (dp_arr[i] > DPLIM) {                                   // check slope of signal
        if (_extu(sig_map[n_max-1],22,22) > i) {                // check endpoint position
          sig_map[n_max-1] &= 0xFFFFFC00;                      // clear old integration endpoint
          sig_map[n_max-1] += (i & 0x3FF);                     // save new integration endpoint
        }
      }
    }                                                           // (e) check for previous pulse

    it = 0;
    while ((dp_arr[i] > DPLIM) && (i < (siglen-1))) {             // rising edge of pulse...
      it++;
      i++;
    }

    if (it > ITLIM-1) {                                          // (b) pulse start found
                                                                // !!!!! 1 + Overflow !!!!!
//M1      sig_map[n_max] = _extu(i-it,20,0);                       // save start index for pulse
      ib1 = i - it;                                             // save start index for pulse

      while ((dp_arr[i] > 0) && (i < (siglen-1))) {               // climb pulse slope...
        i++;
      }

      it  = 0;
      p_a = _extu(p_arr[i], 0,16);                             // get pulse sample (left)
      p_b = _extu(p_arr[i],16,16);                             // get pulse sample (right)
      while (((p_a > OVFL) || (p_b > OVFL)) && (i < (siglen-1))) { // overflows?
        if (p_a > OVFL) iol++;                                   // overflow in left channel?
        if (p_b > OVFL) ior++;                                   // overflow in right channel?
        i++;
        it++;
        p_a = _extu(p_arr[i], 0,16);                           // get next pulse sample (left)
        p_b = _extu(p_arr[i],16,16);                           // get next pulse sample (right)
      }
      oftq_val[n_max] = _extu(iol,24,0) + _extu(ior,16,0);     // save overflow counters
      iol = 0;                                                  // reset overflow counters again
      ior = 0;

      while ((dp_arr[i] >= 0) && (i < (siglen-2))) {              // maximum reached ???
        i++;
      }

      if (i < (siglen-1)) {                                      // save start/max/end points
//M1        sig_map[n_max] += _extu(i-(it/2+it%2),10,0);           // save maximum index for pulse
//M1        sig_map[n_max] += (i-(it/2+it%2)+IQLIM+1);             // save end index for pulse
        ib2 = i - (it/2 + it%2);                                // save maximum index for pulse
        ib3 = i - (it/2 + it%2) + IQLIM + 1;                    // save end index for pulse

        sig_map[n_max] = _extu(ib1,20,0) + _extu(ib2,10,0) + ib3;

        n_max++;                                                // incr. # found peaks
//////////////////////////fast forward
        i += 1;
        while ((dp_arr[i] < 0) && (i < (siglen-1))) {             // "ffwd" to next pulse
          i++;
        }
        i -= 1;
//////////////////////////fast forward
      }

    }                                                           // (e) pulse start found

    i += 1;                                                     // increase loop counter

  }                                                             // (e) loop over signal samples


  if (n_max == 0) {                                           // skip if no signal left
    goto ende;
  }



/************************************/
/* *** start-time determination *** */
/************************************/

```

```
  k = 0;                                                      // reset invalid-signal counter
#pragma MUST_ITERATE(1,ADIM2, );
  for (i=0; i<n_max; i++) {                                    // (b) start time calc.
    loop_a = _extu(sig_map[i], 2,22);                        // signal start sample pos.
    loop_b = _extu(sig_map[i],12,22);                        // signal maximum sample pos.
    p_a    = p_arr[loop_a];                                  // get start sample value
    p_b    = p_arr[loop_b];                                  // get maximum sample value
    p_diff = _sub2((int)p_b,(int)p_a);                       // calc. difference...
    sl     = _ext(p_diff, 0,16);                              // ...extract left difference
    sr     = _ext(p_diff,16,16);                              // ...extract right difference

    if ((sl > 0) && (sr > 0)) {                                 // both slopes positive ?

      stql = 0;
      stqr = 0;
      p_a     = p_arr[loop_a];                               // get start sample value
#pragma MUST_ITERATE(ITLIM-1,ADIM1-1, );
      for (j=loop_a; j<loop_b; j++) {                          // loop from START -> MAX
        p_b      = p_arr[j+1];
        p_diff   = _sub2((int)p_b,(int)p_a);
        stql    += _mpysu(_ext(p_diff, 0,16),j+1);           // left start time
        stqr    += _mpysu(_ext(p_diff,16,16),j+1);           // right start time
        p_a      = p_b;
      }


////////////////////   15.10.02 : do calculations step by step
      stql  = (50*stql)/sl + (int)roff;
      stqr  = (50*stqr)/sr + (int)roff;
      time_val[i-k]  = _ext(stql,16,0) + stqr;
////////////////////
//      stql *= 50;                           // 12.12.2002: problematisch !!!...
//      stqr *= 50;                           // ...Optimizer vertauscht Multiplikation...
//      stql /= sl;                           // ...und Integer-Division!!!
//      stqr /= sr;                           // => weglassen!!!
//      stql += (int)roff;
//      stqr += (int)roff;
//      time_val[i-k] = _ext(stql,16,0) + stqr;
////////////////////  (SO NICHT!!!)
///      time_val[i-k]  = _ext(_sadd(_mpyus(50,stql)/sl,(int)roff),16,0)
///                          + _sadd(_mpyus(50,stqr)/sr,(int)roff);
////////////////////


#ifdef RISETIME
      sum_a          = sl/(loop_b-loop_a);                   // mean slope left
      sum_b          = sr/(loop_b-loop_a);                   // mean slope right
      dpm_val[i-k]   = _ext(sum_a,16,0) + sum_b;
#endif
      if (k > 0) {
        sig_map[i-k]  = sig_map[i];                           // rearrange singal indices
        oftq_val[i-k] = oftq_val[i];                          // rearrange overflow indices
      }
    } else {                                                   // slope(s) not positive...
      k++;                                                    // increase bad signal counter
    }
  }                                                            // (e) start time calc.
  n_max -= k;                                                 // decrease # of valid signals


  if (n_max == 0) {                                           // skip if no signal left
    goto ende;
  }



/**********************************/
/* *** charge determination *** */
/**********************************/

  k = 0;                                                      // reset invalid-signal counter
#pragma MUST_ITERATE(1,ADIM2, );
  for (i=0; i<n_max; i++) {                                    //  (b) start charge analysis
    loop_a = _extu(sig_map[i], 2,22);                        // signal start sample pos.
    loop_b = _extu(sig_map[i],22,22);                        // signal end sample pos.

    if (loop_b > siglen) {                                     // end point out of range?
      loop_b = siglen-1;                                      // set end point to signal length
      sig_map[i]  = _clr(sig_map[i],0,9);                    // erase old end point
      sig_map[i] += loop_b;                                   // save new endpoint
    }

    it = (_extu(sig_map[i],12,22) - loop_a) + 1;             // get # bins for t-int.
    iq = (loop_b - loop_a) + 1;                               // get # bins for Q-int.

    stq = 0;
#pragma MUST_ITERATE(ITLIM+1,ADIM1, );
    for (j=loop_a; j<=loop_b; j++) {                           // integrate charge
      stq = _add2((int)p_arr[j],stq);
    }

    if ((pre_smpl+pst_smpl) > 0) {                            // subtract background
      stq = _sub2(stq,(int)((_extu(bgrdl,16,0)+bgrdr)*iq));
    }
    stql = _ext(stq, 0,16);                                  // extract charge left
    stqr = _ext(stq,16,16);                                  // extract charge right

    if (i > 0) {                                               // were there preceding pulses ?
//M    ib1 = (_extu(sig_map[i-1],22,22)
//M         - _extu(sig_map[i-1],12,22)-1);
      ib2 = (_extu(sig_map[i],2,22)-_extu(sig_map[i-1],22,22));
      if (ib2 > 0) ib2++;
      ib3 = ib1 + ib2;
      if (ib3 < 16 ) {                                         // subtract pre-pulse background
        stql -= (int)_extu(_mpyu(ql_bg,bgc[ib3]),0,7);
        stqr -= (int)_extu(_mpyu(qr_bg,bgc[ib3]),0,7);
      }
    }

    if ((stql > 0) && (stqr > 0)) {
      ib1   = iq - (it + 1);
      stql  = _ext(_mpysu(stql,cor[ib1]),0,7);               // make integration-cutoff
      stqr  = _ext(_mpysu(stqr,cor[ib1]),0,7);               // ...correction for pulse
//      stql  = stql*cor[ib1];                                  // make integration-cutoff
//      stqr  = stqr*cor[ib1];
//      stql  = _ext(stql,0,7);                                // ...correction for pulse
//      stqr  = _ext(stqr,0,7);

      ql_bg = (unsigned int)stql;                              // save pulse background...
      qr_bg = (unsigned int)stqr;                              // ...for next pulse

/*      q_val[i-k]     = _ext(_ext(_mpysu(stql,gainl),0,7),16,0)
                          + _ext(_mpysu(stqr,gainr),0,7);
                          no gain correction!!!
*/

      q_val[i-k]     = _ext(stql,16,0) + stqr;                 // save charge values
      oftq_val[i-k] += _extu(iq,7,0) + it;                     // save it/iq indices

      if (k > 0) {
        time_val[i-k] = time_val[i];                           // rearrange time values
#ifdef RISETIME
        dpm_val[i-k]  = dpm_val[i];                            // rearrange slope values
#endif
        sig_map[i-k]  = sig_map[i];                            // rearrange signal indices
      }
    } else {
      k += 1;                                                   // increase bad signal counter
      ql_bg = 0;                                               // reset saved background
      qr_bg = 0;
    }

  }                                                             //  (e) start charge analysis
  n_max -= k;                                                  // decrease # of valid signals


  if (n_max == 0) {                                            // skip if no signal left
    goto ende;
  }



/************************/
/* *** save results *** */
/************************/

#ifndef RISETIME

#pragma MUST_ITERATE(1,ADIM2, );
  for (i=0; i<n_max; i++) {
    ptr_res[1+i*3] = (unsigned int)time_val[i];                // time values
    ptr_res[2+i*3] = (unsigned int)q_val[i];                   // charge values
    ptr_res[3+i*3] = (unsigned int)oftq_val[i];                // overflow & integration indices
  }
  rlen = n_max*3*L_INT;                                        // calc. data length

#else

#pragma MUST_ITERATE(1,ADIM2, );
  for (i=0; i<n_max; i++) {
    ptr_res[1+i*4] = (unsigned int)time_val[i];             // time values
    ptr_res[2+i*4] = (unsigned int)dpm_val[i];              // (mean) risetime values
    ptr_res[3+i*4] = (unsigned int)q_val[i];                // charge values
    ptr_res[4+i*4] = (unsigned int)oftq_val[i];             // overflow & integration indices
  }
  rlen = n_max*4*L_INT;                                     // calc. data length

#endif
```

```



  ende:
  return rlen;

}
```

D.3.5 Other Programs

Various programs are used for data transfer, communication with SSM and RIO3, consistency checks and for using the DSP timers. They are briefly explained here.

Data Transfer

Transferring data from the GTBRAM to the internal RAM of the DSP is done by using a DMA transfer. For the initialization, source address, destination address and other parameters are written to the corresponding QDMA registers (lines 33–38). The transfer is started by writing the number of words which are to be copied to the QDMA pseudo register (line 40). Although the DSP can execute DMA transfer and other programs in parallel, the program waits until the transfer is finished to ensure that all data is in the internal RAM before the reduction procedure starts (lines 45–48).

```
 1 unsigned int get_gtb_data(unsigned int wcnt, unsigned int blk)
 2 //
 3 // get data from GTBRAM into SAM3 DSP int. RAM
 4 //
 5 //    wcnt : GTBRAM counter data word count
 6 //    blk  : current GTBRAM memory block index
 7 //
 8 {
 9
10 #include "c6211.h"                                               // c6x11 global definitions
11 #include "c6211dma.h"                                            // c6x11 DMA options
12
13 #include "sam3c60res.h"                                          // SAM3 DSP resources
14 #include "sam3vmeres.h"                                          // SAM3 VME resources
15 #include "mem_part.h"                                            // DSP/RIO memory partitioning
16
17 #include "redu_defs.h"                                           // special defs. for data red.
18 #include "redu_glovar_ext.h"                                     // external global variables
19
20
21   unsigned int tmp;                                              // temporary variable
22   unsigned int bcnt;                                             // byte counter for data
23
24
25 // *** clear registers ***
26
27   *(volatile unsigned int *)CIER = 0x00000000;                   // disable all interrupts
28   *(volatile unsigned int *)CIPR = 0x00000001;                   // clear pending interrupts
29
30
31 // *** initiate QDMA transfer ***
32
33   *(int *)QDMA_SRC   = (int)(GTB_DATA_0                          // source address
34                              + 2*L_INT                           // -> event # offset
35                              + blk*(GBLK_SIZE/L_SHORT)*L_INT);   // -> block offset
36   *(int *)QDMA_DST   = (int)DSP_IRAM_D;                          // destination address
37   *(int *)QDMA_OPT   = PRI_H | SIZE_H | DUM_INC | SUM_INDX | TCINT | TCC(0) | FS;
38   *(int *)QDMA_IDX   = 4;
39
40   *(int *)QDMA_S_CNT = (int)(wcnt - 2);                          // start QDMA transfer
41
42
// *** wait for QDMA transfer to finish ***

  tmp = *(volatile unsigned int *)CIPR;                        // wait for QDMA transfer
  while ((tmp & 0x00000001) == 0) {                            // look for 'Rdy' from DMA
    tmp = *(volatile unsigned int *)CIPR;
  }



  bcnt = (wcnt - (2 + 1))*L_SHORT;                             // calc byte counter

  return bcnt;                                                 // return byte counter

}
```

Get SSM Event Number

Comparing the event numbers of SSM and DSP is absolutely necessary in order to guarantee the integrity of the data. The SSM event number is stored in the first four bytes of the active block in the GTBRAM and can be retrieved from there using this program.

```
unsigned int get_ssm_evn(unsigned int blk)
//
// get current event number from SSM
//   (in 'token mode' !!!)
//
//  blk : current active block index for GTBRAM (0,1)
//
{

#include "sam3c60res.h"                                        // SAM3 DSP resources
#include "sam3vmeres.h"                                        // SAM3 VME resources
#include "mem_part.h"                                         // DSP/RIO memory partitioning

#include "redu_defs.h"                                        // special defs. for data red.
#include "redu_glovar_ext.h"                                  // external global variables


           unsigned int  evn;                                  // event number
  volatile unsigned int *p_ram_s;                             // source RAM pointer


// *** calculate source & destination RAM pointers ***

  p_ram_s = (volatile unsigned int *)(GTB_DATA_0
                           + blk*(GBLK_SIZE/L_SHORT)*L_INT);   // calc. GTBRAM source pointer


// *** copy data from GTBRAM ***

  evn  =  p_ram_s[0] & 0xffff;                                // get 'lower' 16 bit...
  evn |= (p_ram_s[1] & 0xffff) << 16;                           // ...and 'upper' 16 bit


// *** return variable ***

  return evn;                                                  // return SSM event number

}
```

Compare Event Numbers

This program compares SSM and DSP event numbers and, in case of a mismatch, returns an error message to the DSP.

```
unsigned int comp_evn(unsigned int evndsp, unsigned int evnssm, unsigned int ib)
//
// compare DSP and SSM event numbers...
//
//    evndsp : DSP event number
//    evnssm : SSM event number
//    ib     : active memory block index: 0/1
//
// returns error flag:   0  -> no errors
//                    else -> errors occured
//
{

#include "sam3c60res.h"                                      // SAM3 DSP resources
#include "sam3vmeres.h"                                      // SAM3 VME resources
#include "mem_part.h"                                        // DSP/RIO memory partitioning


  unsigned int xtmp;                                         // temporary variable


// *** compare & read back event numbers ***

  if (evndsp != evnssm) {

    return (unsigned int)COM_Error_BadEvNo;                 // set error pattern

  } else {

    xtmp = *(volatile unsigned int *)(DSP_OPT_DE0
                                      + ib*BOPT_OFFS);       // get DSP evt. #
    if (xtmp != evndsp) {
      return (unsigned int)COM_Error_EvNoWE;                // set error pattern
    }

    xtmp = *(volatile unsigned int *)(DSP_OPT_SE0
                                      + ib*BOPT_OFFS);       // get SSM evt. #
    if (xtmp != evnssm) {
      return (unsigned int)COM_Error_EvNoWE;                // set error pattern
    }

  }

  return 0x00000000;                                         // return error flag

}
```

Timer Information

The DSP contains two independent timers which for example can be used to determined the speed of a DMA transfer or of the reduction procedure. This program allows to reset, start and stop both timers. The timers are counting at a quarter of the CPU clock frequency.

```
unsigned int timer_ctrl(unsigned int i_tim, unsigned int i_ctl)
//
// control timer 0 & 1 (start/stop/evaluate)
//
//    i_tim : timer number...
//             0 : timer 0
//             1 : timer 1
//    i_ctl : timer function control...
//             0 : reset & start
//             1 : stop
//
{

#include "c6211.h"                                           // c6x11 global definitions


#define  TOUT   0                                            // activate TOUT pin (0,1)
#define  GO     6                                            // reset/start timer counter (1)
#define  NHLD   7                                            // disable/enable counter(0,1)
#define  CLKSRC 9                                            // clock source ext./CPU (0,1)



          unsigned int  tcnt;                                // time counter



// *** select timer & execute function ***

  tcnt = 0;                                                  // reset timer counter
  if (i_tim == 0) {
    if (i_ctl == 0) {
      *(volatile unsigned int *)TIMER0_PRD   = 0x001fffff;     // timer period
      *(volatile unsigned int *)TIMER0_COUNT = 0x00000000;     // reset counter
//      *(volatile unsigned int *)TIMER0_CTRL  = (0 << TOUT)     // -> TOUT pin
//                                             | (1 << GO)       // -> GO: look for !HLD
//                                             | (1 << NHLD)     // -> !HLD: counter may count now
//                                             | (1 << CLKSRC);  // -> CLKSRC: CPU clock/4
      *(volatile unsigned int *)TIMER0_CTRL  = (0 << TOUT)     // -> TOUT pin
                                             | (1 << CLKSRC);  // -> CLKSRC: CPU clock/4
      *(volatile unsigned int *)TIMER0_CTRL |= (1 << GO)       // -> GO: look for !HLD
                                             | (1 << NHLD);    // -> CLKSRC: CPU clock/4
    } else if (i_ctl == 1) {
      *(volatile unsigned int *)TIMER0_CTRL  = (0 << GO)       // -> GO:   stop counter
                                             | (0 << NHLD);    // -> !HLD: disable counter
      tcnt = *(volatile unsigned int *)TIMER0_COUNT;
    }
  } else if (i_tim == 1) {
    if (i_ctl == 0) {
      *(volatile unsigned int *)TIMER1_PRD   = 0x001fffff;     // timer period
      *(volatile unsigned int *)TIMER1_COUNT = 0x00000000;     // reset counter
//      *(volatile unsigned int *)TIMER1_CTRL  = (0 << TOUT)     // -> TOUT pin
//                                             | (1 << GO)       // -> GO: look for !HLD
//                                             | (1 << NHLD)     // -> !HLD: counter may count now
//                                             | (1 << CLKSRC);  // -> CLKSRC: CPU clock/4
      *(volatile unsigned int *)TIMER1_CTRL  = (0 << TOUT)     // -> TOUT pin
                                             | (1 << CLKSRC);  // -> CLKSRC: CPU clock/4
      *(volatile unsigned int *)TIMER1_CTRL |= (1 << GO)       // -> GO: look for !HLD
                                             | (1 << NHLD);    // -> CLKSRC: CPU clock/4
    } else if (i_ctl == 1) {
      *(volatile unsigned int *)TIMER1_CTRL  = (0 << GO)       // -> GO:   stop counter
                                             | (0 << NHLD);    // -> !HLD: disable counter
      tcnt = *(volatile unsigned int *)TIMER1_COUNT;
    }
  }


// *** return variable ***

  return tcnt;                                                 // return byte counter

}
```

Communication with RIO3

DSP and RIO3 use a small part of the VRAM for their communication. This program is used to write the current status of the main program to the corresponding memory addresses in the VRAM.

```
void mess_stat_blk(unsigned int stat, unsigned int blk)
//
// write program status word to global/local control memory section
//
//    stat : new program status
//    blk  : current active memory block (GTBRAM/VRAM)
//
{

#include "sam3c60res.h"                                        // SAM3 DSP resources
#include "sam3vmeres.h"                                        // SAM3 VME resources
#include "mem_part.h"                                          // DSP/RIO memory partitioning


// *** set mode, next block & 'Ack' (address mode !!!) ***

  *(volatile unsigned int *)(DSP_OPT_CTL)                = stat;  // signal to RIO
  *(volatile unsigned int *)(DSP_OPT_ST0 + blk*BOPT_OFFS) = stat;

}
```

Communication with SSM

When the data of an event have been copied from the GTBRAM, the sampling of the FADCs is started again. This is done by writing an 'acknowledge' bit to the GTB control register. In addition, the start address for the data of the coming event is set to the next free memory block in the GTBRAM.

```
void send_ack_tm(unsigned int blk)
//
// set next (!) active memory block number and
// send 'Ack' to Scanner (SSM)
//
//   blk : current memory block number
//
//   => for token-mode programs
//
{

#include "sam3c60res.h"                                       // SAM3 DSP resources
#include "sam3vmeres.h"                                       // SAM3 VME resources
#include "mem_part.h"                                         // DSP/RIO memory partitioning

#include "redu_defs.h"                                        // definitions f. data red.
#include "redu_glovar_ext.h"                                  // external (global) variables


//      unsigned int tmp1;

// *** set mode, next block & 'Ack' (token mode !!!) ***

//  *(volatile unsigned int *)DSP_GTBXADRPTR = 0;               // set GTBRAM cnt. to next block
  *(volatile unsigned int *)DSP_GTBXADRPTR =
                           ((blk+1)%2)*(GBLK_SIZE/L_SHORT);    // set GTBRAM cnt. to next block

/////////////////////////// prevent 'Ack' to be set too early !!!
//    tmp1 = *(volatile unsigned int *)DSP_GTBXADRPTR;
//    while (tmp1 != (((blk+1)%2)*(GBLK_SIZE/L_SHORT))) {
//      *(volatile unsigned int *)DSP_GTBXADRPTR =
//                             ((blk+1)%2)*(GBLK_SIZE/L_SHORT);
//      tmp1 = *(volatile unsigned int *)DSP_GTBXADRPTR;
//    }
///////////////////////////

  *(volatile unsigned int *)DSP_GTBXDSR    = DSR_ACK;         // set 'Ack' for SSM

}
```

D.4 Further Programs

Further programs for various purposes exist which are not used in the reduction program presented here:

- `pushthrough.c`: This program adds another option of how to read out the raw data. In addition to just copying the raw data (mode: RAW) or to reduce it (mode: RED) it acts as a filter which rejects hitlets below a certain length. The remaining data is not processed further and copied into the VRAM (mode: PTR).

- `dma_data_irvr.c`: Whenever using the RAW mode this program is needed in order to copy the raw data from the internal RAM of the DSP into the VRAM via a DMA transfer.

- `get_hbuf_data.c`: Reading data directly from the SSM's hitbuffer in the address mode ('_am') into the VRAM or the internal RAM of the DSP can be don with this program.

- `mess_stat_all.c`: If no partitioning of GTBRAM and VRAM is needed (blocks 0,1), the program status of the DSP program is reported to the RIO3 using this program. It will write the current status word to the communication block and to both data block headers in the VRAM.

- `send_ack_am.c`: Intended for programs using the address mode instead of the token mode for the data transport between SSM and SAM3. This program sends the 'acknowledge' signal which restarts the SSM after reading the data and sets the GTBRAM address pointer to the next free memory block.

- `display_text.c`, `blink_led.c`, `i_to_a.c` and `mysleep.c`: These programs were written for debugging purposes. They allow to control the LEDs of the SAM3 module and to write text to its dot-matrix display.

D.5 Generating executable DSP Programs

Executable programs for the DSP can be created with the 'Code Composer Studio' software provided by Texas Instruments. In addition to the program source code and the include files, reset/interrupt vectors and the memory partition of the DSP have to be defined. Special functions for the respective DSP type have to be included from a program library. Subsequently, an executable DSP program is compiled and a program image file is generated. This file can then be loaded into the DSP memory and the program can be started.

D.5.1 Reset and Interrupt Vectors

The reset/interrupt vectors defined in the file `vectors.asm` are needed by the DSP in order to react to a 'reset' signal or to other hardware interrupts. In our case no hardware interrupts are used. If a 'reset' signal is given, the DSP will start to execute the program which was loaded into its internal RAM (lines 40–48).

```
 1 ;
 2 ;  Copyright 2001 by Texas Instruments Incorporated.
 3 ;  All rights reserved. Property of Texas Instruments Incorporated.
 4 ;  Restricted rights to use, duplicate or disclose this code are
 5 ;  granted through contract.
 6 ;  U.S. Patent Nos. 5,283,900  5,392,448
 7 ;
 8 ; "@(#) DSP/BIOS 4.51.0 05-23-01 (barracuda-i10)"
 9 ;
10 ;  ======== vectors.asm ========
11 ;  Plug in the entry point at RESET in the interrupt vector table
12 ;
13
14 ;
15 ;  ======== unused ========
16 ;  plug inifinite loop -- with nested branches to
```

```
;  disable interrupts -- for all undefined vectors
;
unused  .macro id

        .global unused:id:
unused:id:
        b unused:id:    ; nested branches to block interrupts
        nop 4
        b unused:id:
        nop
        nop
        nop
        nop
        nop

        .endm

        .sect ".vectors"

        .ref _c_int00          ; C entry point

        .align  32*8*4          ; must be aligned on 256 word boundary

RESET:                          ; reset vector
        mvkl _c_int00,b0       ; load destination function address to b0
        mvkh _c_int00,b0
        b b0                    ; start branch to destination function
        mvc PCE1,b0             ; address of interrupt vectors
        mvc b0,ISTP             ; set table to point here
        nop 3                   ; fill delay slot
        nop
        nop

        ;
        ;  plug unused interrupts with infinite loops to
        ;  catch stray interrupts
        ;
        unused 1
        unused 2
        unused 3
        unused 4
        unused 5
        unused 6
        unused 7
        unused 8
        unused 9
        unused 10
        unused 11
        unused 12
        unused 13
        unused 14
        unused 15
```

D.5.2 DSP Memory Partition

The C6711 DSP has an internal memory of 32bit×16k (= 64 kB) which is used for both, program code and data. Using the file `sam3.cmd` the partition of this memory is defined and different parts of the DSP program are assigned to different parts of the memory.

Here, the section MEMORY defines two partitions inside the internal RAM called 'VECS' and 'IRAM'. The 'VECS' partition starts at memory address 0 and has a length of 0x220 (= 544 B). It is reserved for the reset and interrupt vectors. All other parts of the DSP program are placed into the 'IRAM' partition which starts at 0x220 with a length of 0x7DDC (= 32220 B). Therefore, with this partitioning 32 kB of the internal memory remain free for the drift chamber data. The file `sam3_48k.cmd` is used to reserve 48 kB of the internal RAM for drift chamber data.

```
-stack 0x200
-heap 0x200


MEMORY
{
  VECS:       org=          0h,  len=       0x220
/*  IRAM:       org = 0x00000220,  len = 0x0000DDE0 */ /* int. RAM a la Wolfgang */
/*  IRAM:       org = 0x00000220,  len = 0x0000FFFC */ /* full internal RAM */
/*  IRAM:       org = 0x00000220,  len = 0x00007FFC */
  IRAM:       org = 0x00000220,  len = 0x00007DDC

}

SECTIONS
{
  /* Created in vectors.asm */
  .vectors  :    > VECS

  /* Created by C Compiler  */
  .text       :    > IRAM
  .bss     :    > IRAM
   coeffs  :    > IRAM
  .cinit   :    > IRAM
  .stack   :    > IRAM
  .sysmem  :    > IRAM
  .const   :    > IRAM
  .switch  :    > IRAM
  .far     :    > IRAM
  .cio     :    > IRAM

}
```

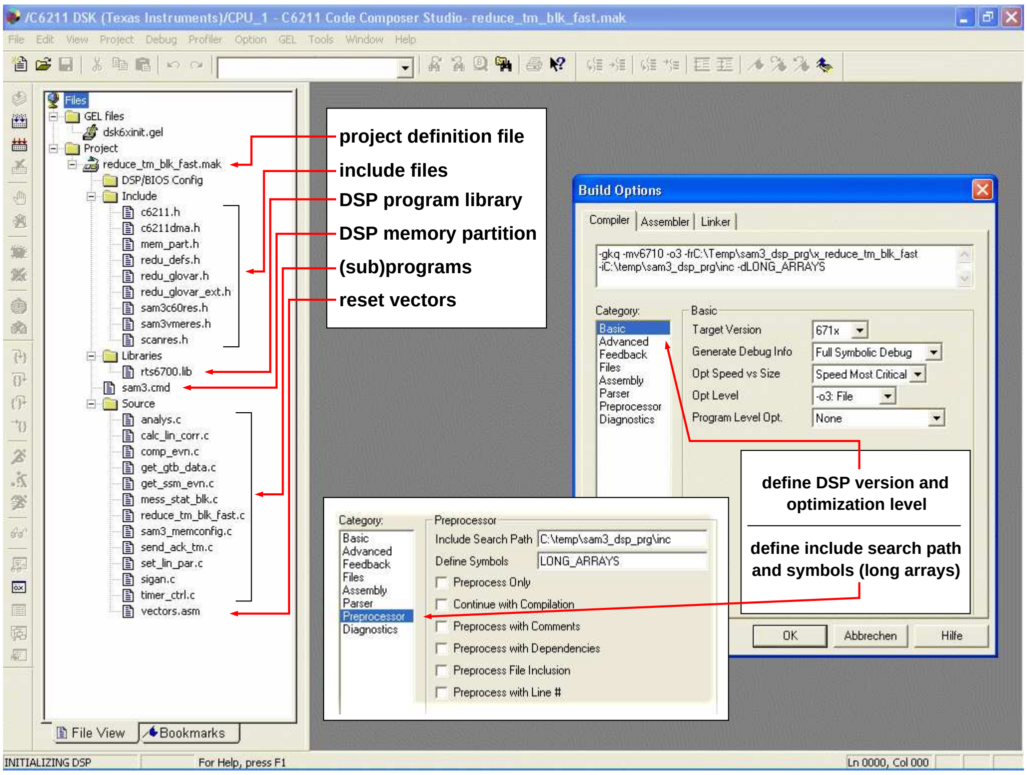
D.5.3 Building and Compiling a Project

Several steps have to be taken in order to build a project for a DSP program illustrated in figure D.2. The project definition file ('.mak') contains information about all files which are included into the project and about all other options. The files which are needed for the project are the include files, the program library for the DSP, the memory partition file ('.cmd'), the source code of main program and sub-programs and the definitions of the reset and interrupt vectors. Furthermore, the DSP type and the optimization level should be specified in the build options (basic). For the preprocessor, the path of the include files and definitions of special symbols have to be set. Afterwards, the DSP program can be compiled and a binary output file ('.out') is generated.

D.5.4 Producing a DSP Program Image

The DSP cannot directly execute the binary output file. It has to be converted into a program image file ('.m0') first, which afterwards is loaded into the internal memory of the DSP. This file is produced with a small program called 'hex6x'. The syntax is

```
hex6x -m [pname] -o [pname].m0 -map [pname].mmx...
                         ...-memwidth 32 -romwidth 32
```



```
-gkq -mv6710 -o3 -frC:\Temp\sam3_dsp_prg\x_reduce_tm_blk_fast
-iC:\temp\sam3_dsp_prg\inc -dLONG_ARRAYS
```

Basic

- Target Version: 671x
- Generate Debug Info: Full Symbolic Debug
- Opt Speed vs Size: Speed Most Critical
- Opt Level: -o3: File
- Program Level Opt.: None

Preprocessor

- Include Search Path: C:\temp\sam3_dsp_prg\inc
- Define Symbols: LONG_ARRAYS
- ☐ Preprocess Only
- ☐ Continue with Compilation
- ☐ Preprocess with Comments
- ☐ Preprocess with Dependencies
- ☐ Preprocess File Inclusion
- ☐ Preprocess with Line #

Figure D.2: Illustration of a project for a DSP program in the 'Code Composer Studio'.

where '[pname]' denotes the name of the DSP program. Once the program is in the DSP memory it can be started by sending a 'reset' signal.

Appendix E

Analysis Cuts

Different sets of cuts which were used in the strange particle reconstruction are documented here. The following tables are organized such that the column '**class**' denotes the subgroups of cuts, e.g. vertex cuts (global), cuts for the decay particles (π^-, π^+ and proton) and cuts for the mother particles (K^0, Λ^0). Each cut quantity is given with its unit and the meaning is explained. The rightmost columns contain the specific cut thresholds or cut ranges for a specific cut set.

Table E.1 presents the final K^0 reconstruction cuts. Two sets named '**1**' and '**2**' exist and the first one was used in the analysis. The most important differences between the cut sets are the lower hit-multiplicity (HMUL) thresholds for the decay particles and the track quality cuts on σ_{dxy}. Differences between other quantities have a minor effect on the signal quality.

The cut for the Λ^0 analysis are summarized in table E.2. Out of three sets named '**1**', '**2**' and '**3**' the first one was finally used. The different signal-to-background ratios (cf. table 5.4) are mostly due to different cuts on track quality of the decay particles (σ_{dxy}), transverse flight path of the mother particle (r_t) and on the pointing angle ($|\Delta\phi|$).

For completeness, the $\Sigma^-(1385)$ and Ξ^- reconstruction cuts are provided in table E.3. These cuts are by no means the final ones.

quantity	unit	class	meaning	cut set	
				1 (MM)	**2 (NH)**
$\|v_z\|$	[cm]	**global**	z coord. of vertex	$< 10.$	$< 10.$
$\|v_x\|$	[cm]		x coord. of vertex	< 1.5	< 1.5
$\|v_y\|$	[cm]		y coord. of vertex	< 1.5	< 1.5
PMUL	—		PLAWA mult.	> 0	> 0
$\|d_0\|$	[cm]	π^-	dist.-to-vtx. in x-y	$1.5 < \cdots < 20.$	$1.5 < \cdots < 20.$
HMUL	—		# of hits per track	> 24	> 30
p_t	[GeV/c]		transv. momentum	> 0.09	> 0.12
m_{CDC}	[GeV/c^2]		particle mass	$0.06 < \cdots < 0.6$	$0.04 < \cdots < 0.7$
$\|z_0\|$	[cm]		dist.-to-vtx. in r-z	$< 30.$	$< 50.$
σ_{dxy}	[cm]		track quality in x-y	< 0.5	< 0.08
σ_{z_0}	[cm]		track quality in z	$< 50.$	$< 50.$
$\|d_0\|$	[cm]	π^+	dist.-to-vtx. in x-y	$0.9 < \cdots < 20.$	$0.9 < \cdots < 20.$
HMUL	—		# of hits per track	> 24	> 25
p_t	[GeV/c]		transv. momentum	> 0.09	> 0.12
m_{CDC}	[GeV/c^2]		particle mass	$0.06 < \cdots < 0.6$	$0.04 < \cdots < 0.7$
$\|z_0\|$	[cm]		dist.-to-vtx. in r-z	$< 30.$	$< 50.$
σ_{dxy}	[cm]		track quality in x-y	< 0.5	< 0.08
σ_{z_0}	[cm]		track quality in z	$< 50.$	$< 50.$
r_t	[cm]	K^0	transv. flight path	$1.5 < \cdots < 20.$	$1.8 < \cdots < 35.$
Δz	[cm]		z dist. of tracks	$< 60.$	$< 60.$
$\|d_0\|$	[cm]		dist.-to-vtx. in x-y	< 1.5	< 0.7
$\|\Delta\phi\|$	[°]		pointing angle	$< 11.$	$< 15.$
p_t	[GeV/c]		transv. momentum	> 0.05	> 0.01
$\|z_0\|$	[cm]		dist.-to-vtx. in r-z	$< 100.$	$< 100.$
$m_{K_S^0}$	[GeV/c^2]		particle mass	$0.35 < \cdots < 0.8$	$0.3 < \cdots < 0.8$

Table E.1: Final cuts for the K_S^0 analysis. Two sets of cuts (1,2) are available, the one used in the analysis was the '1' set.

				cut set		
quantity	**unit**	**class**	**meaning**	**1 (MMRK)**	**2 (NH_P)**	**3 (NH_S)**
$\lvert v_z \rvert$	[cm]	**global**	z coord. of vertex	$< 10.$	$< 10.$	$< 10.$
$\lvert v_x \rvert$	[cm]		x coord. of vertex	< 1.5	< 1.5	< 1.5
$\lvert v_y \rvert$	[cm]		y coord. of vertex	< 1.5	< 1.5	< 1.5
PMUL	—		PLAWA mult.	> 0	> 0	> 0
$\lvert d_0 \rvert$	[cm]	π^-	dist.-to-vtx. in x-y	$2.0 < \cdots < 40.$	$2.0 < \cdots < 20.$	$2.0 < \cdots < 20.$
HMUL	—		# of hits per track	> 24	> 24	> 24
p_t	[GeV/c]		transv. momentum	> 0.10	> 0.09	> 0.09
m_{CDC}	[GeV/c^2]		particle mass	$0.05 < \cdots < 0.7$	$0.05 < \cdots < 0.7$	$0.08 < \cdots < 0.7$
$\lvert z_0 \rvert$	[cm]		dist.-to-vtx. in r-z	$< 30.$	$< 30.$	$< 30.$
σ_{dxy}	[cm]		track quality in x-y	< 0.5	< 0.5	< 0.1
σ_{z_0}	[cm]		track quality in z	$< 50.$	$< 50.$	$< 50.$
$\lvert d_0 \rvert$	[cm]	**proton**	dist.-to-vtx. in x-y	$0.65 < \cdots < 40.$	$0.55 < \cdots < 20.$	$0.6 < \cdots < 20.$
HMUL	—		# of hits per track	> 29	> 24	> 29
p_t	[GeV/c]		transv. momentum	> 0.10	> 0.25	> 0.3
m_{CDC}	[GeV/c^2]		particle mass	$0.5 < \cdots < 1.5$	$0.7 < \cdots < 1.5$	$0.7 < \cdots < 1.5$
$\lvert z_0 \rvert$	[cm]		dist.-to-vtx. in r-z	$< 30.$	$< 30.$	$< 30.$
σ_{dxy}	[cm]		track quality in x-y	< 0.5	< 0.5	< 0.05
σ_{z_0}	[cm]		track quality in z	$< 50.$	$< 50.$	$< 50.$
r_t	[cm]	Λ^0	transv. flight path	$2. < \cdots < 20.$	$3. < \cdots < 30.$	$4. < \cdots < 30.$
Δz	[cm]		z dist. of tracks	$< 60.$	$< 50.$	$< 20.$
$\lvert d_0 \rvert$	[cm]		dist.-to-vtx. in x-y	< 0.5	< 0.5	< 0.5
$\lvert \Delta\phi \rvert$	[°]		pointing angle	$< 4.$	$< 4.$	$< 2.$
p_t	[GeV/c]		transv. momentum	> 0.3	> 0.3	> 0.3
$\lvert z_0 \rvert$	[cm]		dist.-to-vtx. in r-z	$< 100.$	$< 100.$	$< 100.$
m_{Λ^0}	[GeV/c^2]		particle mass	$1. < \cdots < 1.25$	$1. < \cdots < 1.25$	$1. < \cdots < 1.25$

Table E.2: Final cuts for the Λ^0 analysis. From the three different set of cuts (1,2,3) the '1' set was used in the final analysis.

quantity	unit	class	meaning	cut set	
				Ξ^-	$\Sigma^-(1385)$
$\lvert v_z \rvert$	[cm]	**global**	z coord. of vertex	$< 10.$	$< 10.$
$\lvert v_x \rvert$	[cm]		x coord. of vertex	< 1.5	< 1.5
$\lvert v_y \rvert$	[cm]		y coord. of vertex	< 1.5	< 1.5
PMUL	—		PLAWA mult.	> 0	> 0
$\lvert d_0 \rvert$	[cm]	1^{st} π^-	dist.-to-vtx. in x-y	$1.2 < \cdots < 20.$	$1.9 < \cdots < 20.$
HMUL	—		# of hits per track	> 24	> 24
p_t	[GeV/c]		transv. momentum	> 0.10	> 0.09
m_{CDC}	[GeV/c^2]		particle mass	$0.05 < \cdots < 0.7$	$0.08 < \cdots < 0.7$
$\lvert z_0 \rvert$	[cm]		dist.-to-vtx. in r-z	$< 100.$	$< 100.$
σ_{dxy}	[cm]		track quality in x-y	< 0.5	< 0.1
σ_{z_0}	[cm]		track quality in z	$< 50.$	$< 50.$
$\lvert d_0 \rvert$	[cm]	**proton**	dist.-to-vtx. in x-y	$0.5 < \cdots < 5.$	$0.6 < \cdots < 20.$
HMUL	—		# of hits per track	> 24	> 29
p_t	[GeV/c]		transv. momentum	> 0.30	> 0.3
m_{CDC}	[GeV/c^2]		particle mass	$0.5 < \cdots < 1.5$	$0.7 < \cdots < 1.5$
$\lvert z_0 \rvert$	[cm]		dist.-to-vtx. in r-z	$< 100.$	$< 100.$
σ_{dxy}	[cm]		track quality in x-y	< 0.5	< 0.05
σ_{z_0}	[cm]		track quality in z	$< 50.$	$< 50.$
r_{t,Λ^0}	[cm]	Λ^0	transv. flight path	$6. < \cdots < 35.$	$4. < \cdots < 30.$
Δz	[cm]		z dist. of tracks	$< 28.$	$< 20.$
$\lvert d_0 \rvert$	[cm]		dist.-to-vtx. in x-y	$< 5.$	< 0.5
$\lvert \Delta\phi \rvert$	[°]		pointing angle	$< 20.$	$< 2.$
p_t	[GeV/c]		transv. momentum	$> 0.$	> 0.3
$\lvert z_0 \rvert$	[cm]		dist.-to-vtx. in r-z	$< 1000.$	$< 100.$
m_{Λ^0}	[GeV/c^2]		particle mass	$1.106 < \cdots < 1.126$	$1. < \cdots < 1.25$
$\lvert d_0 \rvert$	[cm]	2^{nd} π^-	dist.-to-vtx. in x-y	$0.3 < \cdots < 10.$	$0. < \cdots < 0.7$
HMUL	—		# of hits per track	> 24	> 24
p_t	[GeV/c]		transv. momentum	> 0.10	> 0.10
m_{CDC}	[GeV/c^2]		particle mass	$0.05 < \cdots < 0.7$	$0.05 < \cdots < 0.5$
$\lvert z_0 \rvert$	[cm]		dist.-to-vtx. in r-z	$< 100.$	$< 100.$
σ_{dxy}	[cm]		track quality in x-y	< 0.5	< 0.06
σ_{z_0}	[cm]		track quality in z	$< 50.$	$< 50.$
r_{t,Ξ^-}	[cm]	Ξ^-	transv. flight path	$0. < \cdots < 15.$	$0. < \cdots < 1.0$
Δz	[cm]		z dist. of tracks	$< 100.$	$< 100.$
$\lvert d_0 \rvert$	[cm]		dist.-to-vtx. in x-y	< 0.5	< 1.0
$\lvert \Delta\phi \rvert$	[°]		pointing angle	$< 15.$	$< 180.$
p_t	[GeV/c]		transv. momentum	$> 0.$	$> 0.$
$\lvert z_0 \rvert$	[cm]		dist.-to-vtx. in r-z	$< 1000.$	$< 1000.$
corr. r_{t,Λ^0}	[cm]		corr. transv. fp.	$0. < \cdots < 20.$	$0. < \cdots < 35.$

Table E.3: Cuts for the Ξ^- and the $\Sigma^-(1385)$ analysis.

Bibliography

[1] R. Stock. Particle production in high energy nucleus-nucleus collisions. *Phys. Rept.*, 135:259–315, 1986.

[2] T. Hatsuda. Theoretical overview: Hot and dense QCD in equilibrium. *Nucl. Phys.*, A544:27–48, 1992.

[3] P. Braun-Munzinger and J. Stachel. Particle Ratios, Equilibration, and the QCD Phase Boundary. *J. Phys. G*, 28:1971–1976, 2002.

[4] P. Braun-Munzinger, J. Stachel, J.P. Wessels, and N. Xu. Thermal Equilibration and Expansion in Nucleus-Nucleus Collisions at the AGS. *Phys. Lett.*, B344:43–48, 1995.

[5] P. Braun-Munzinger, J. Stachel, J.P. Wessels, and N. Xu. Thermal and Hadrochemical Equilibration in Nucleus-Nucleus Collisions at the SPS. *Phys. Lett.*, B365:1–6, 1995.

[6] P. Braun-Munzinger and J. Stachel. Dynamics ofUltra-Relativistic Nuclear Collisions with Heavy Beams: An Experimental Overview. *Nucl. Phys.*, A638:3–18, 1998.

[7] P. Braun-Munzinger, I. Heppe, and J. Stachel. Chemical Equilibration in Pb+Pb collisions at the SPS. *Phys. Lett.*, B465:15–20, 1999.

[8] P. Braun-Munzinger, D. Magestro, K. Redlich, and J. Stachel. Hadron Production in Au+Au collisions at RHIC. *Phys. Lett.*, B518:41–46, 2001.

[9] J.W. Harris and B. Müller. The search for the quark-gluon plasma. *Annu. Rev. Nucl. Part. Sci.*, 46:71–107, 1996.

[10] G.E. Brown and M. Rho. Chiral restoration in hot and/or dense matter. *Phys. Rept.*, 269:333–380, 1996.

[11] V. Koch. Aspects of Chiral Symmetry. *Int. J. Mod. Phys.*, E6:203–250, 1997.

[12] H. Stöcker and W. Greiner. High energy heavy ion collisions: Probing the equation of state of highly excited hadronic matter. *Phys. Rept.*, 137:277–392, 1986.

[13] J. Aichelin and Che Ming Ko. Subthreshold Kaon Production as a Probe of the Nuclear Equation of State. *Phys. Rev. Lett.*, 55:2661–2663, 1985.

[14] C. Sturm et al. Evidence for a Soft Nuclear Equation-of-State from Kaon Production in Heavy Ion Collisions. *Phys. Rev. Lett.*, 86:39–42, 2001.

[15] C. Fuchs, Amand Faessler, S. El-Basaouny, and E. Zabrodin. The nuclear equation of state probed by K^+ production in heavy ion collisions. *J. Phys. G*, 28:1615–1622, 2002.

[16] Y. Nambu and G. Jona-Lasinio. Dynamical Model of Elementary Particles Based on an Analogy with Superconductivity. i. *Phys. Rev.*, 122:345–358, 1961.

[17] Y. Nambu and G. Jona-Lasinio. Dynamical Model of Elementary Particles Based on an Analogy with Superconductivity. ii. *Phys. Rev.*, 124:246–254, 1961.

[18] W. Weise. Chiral Dynamics in Nuclear Systems. *Prog. Theor. Phys. Suppl.*, 149:1–19, 2003.

[19] M. Gell-Mann, R.J. Oakes, and B. Renner. Behavior of Current Divergences under $\mathrm{SU}_3 \times \mathrm{SU}_3$. *Phys. Rev.*, 175:2195–2199, 1968.

[20] G.E. Brown and Mannque Rho. Scaling Effective Lagrangians in a Dense Medium. *Phys. Rev. Lett.*, 66:2720–2723, 1991.

[21] T. Hatsuda and S.H. Lee. QCD sum rules for vector mesons in the nuclear medium. *Phys. Rev.*, C46:34–38, 1992.

[22] H. Kuwabara and T. Hatsuda. ϕ-meson in Nuclear Matter. *Prog. Theor. Phys.*, 94:1163–1168, 1995.

[23] F. Klingl, N. Kaiser, and W. Weise. Current correlation functions, QCD sum rules and vector mesons in baryonic matter. *Nucl. Phys.*, A624:527–563, 1997.

[24] B. Friman. Vector meson propagation in dense matter. In *Proc. APCTP Workshop on Astro-Hadron Physics, Seoul, Korea, October 25-31*, pages 337–354, 1997.

[25] F. Klingl, T. Waas, and W. Weise. Modification of the ϕ-meson spectrum in nuclear matter. *Phys. Lett.*, B431:254–262, 1998.

[26] W. Weise. Hadrons in Dense Baryonic Matter. *Nucl. Phys.*, A610:35c–48c, 1996.

[27] J. Schaffner, A. Gal, I.N. Mishustin, H. Stöcker, and W. Greiner. Kaon effective mass and energy in dense nuclear matter. *Phys. Lett.*, B334:268–274, 1994.

[28] C.H. Lee. Kaon condensation in dense stellar matter. *Phys. Rept.*, 275:255–341, 1996.

[29] G.Q. Li and C.M. Ko. Kaon flow in heavy-ion collisions. *Nucl. Phys.*, A594:460–482, 1995.

[30] J. Schaffner, J. Bondorf, and I.N. Mishustin. In-medium Production of Kaons at the Mean-Field Level. *Nucl. Phys.*, A625:325–346, 1997.

[31] D.B. Kaplan and A.E. Nelson. Strange goings on in dense nucleonic matter. *Phys. Lett.*, B175:57–63, 1986.

[32] A.E. Nelson and D.B. Kaplan. Strange condensate realignment in relativistic heavy ion collisions. *Phys. Lett.*, B192:193–197, 1987.

[33] G.E. Brown, K. Kubodera, D. Page, and P. Pizzochero. Strangeness condensation and cooling of neutron stars. *Phys. Rev.*, D37:2042–2046, 1988.

[34] G.E. Brown and H. Bethe. A scenario for a large number of low mass black holes in the galaxy. *Astrophys. J.*, 423:659, 1994.

[35] G.E. Brown, C.-H. Lee, Mannque Rho, and V. Thorsson. From kaon-nuclear interactions to kaon condensation. *Nucl. Phys.*, A567:937–956, 1994.

[36] G.-Q. Li, C.-H. Lee, and G.E. Brown. Kaons in dense matter, kaon production in heavy-ion collisions, and kaon condensation in neutron stars. *Nucl. Phys.*, A625:372–434, 1997.

[37] P. Danielewicz. Flow and equation of state in heavy-ion collisions. *Nucl. Phys.*, A661:82–92, 1999.

[38] G.D. Westfall et al. Nuclear Fireball Model for Proton Inclusive Spectra from Relativistic Heavy-Ion Collisions. *Phys. Rev. Lett.*, 37:1202–1205, 1976.

[39] J. Gosset et al. Central collisions of relativistic heavy ions. *Phys. Rev.*, C16:629–657, 1977.

[40] F. Rami et al. Isospin Tracing: A Probe of Nonequilibrium in Central Heavy-Ion Collisions. *Phys. Rev. Lett.*, 84:1120–1123, 2000.

[41] W. Reisdorf et al. Nuclear Stopping from 0.09A to 1.93A GeV and Its Correlation to Flow. *Phys. Rev. Lett.*, 92:232301, 2004.

[42] W. Reisdorf and H.G. Ritter. Collective flow in heavy-ion collisions. *Ann. Rev. Nucl. Part. Sci.*, 47:663–709, 1997.

[43] N. Herrmann for the FOPI collaboration. Particle Production and Flow at SIS Energies. *Nucl. Phys.*, A610:49c–62c, 1996.

[44] N. Herrmann for the FOPI collaboration. Strangeness Production and Propagation in Relativistic Heavy Ion Collisions at SIS Energies. *Prog. Part. Nucl. Phys.*, 42:187–196, 1999.

[45] P. Senger for the KaoS collaboration. Strange Mesons as a Probe for Dense Nuclear Matter. *Prog. Part. Nucl. Phys.*, 42:209–219, 1999.

[46] H. Oeschler. Survey of experimental data. *J. Phys. G*, 28:1787–1800, 2002.

[47] F. Laue et al. Medium Effects in Kaon and Antikaon Production in Nuclear Collisions at Subthreshold Beam Energies. *Phys. Rev. Lett.*, 82:1640–1643, 1999.

[48] R. Barth et al. Subthreshold Production of Kaons and Antikaons in Nucleus-Nucleus Collisions at Equivalent Beam Energies. *Phys. Rev. Lett.*, 78:4007–4010, 1997.

[49] M. Menzel et al. First Measurement of Antikaon Phase-Space Distributions in Nucleus-Nucleus Collisions at Subthreshold Beam Energies. *Phys.Lett.*, B495:26–32, 2000.

[50] D. Best et al. K^+ production in the reaction ^{58}Ni+^{58}Ni at incident energies from 1 to 2 AGeV. *Nucl.Phys.*, A625:307–324, 1997.

[51] C. Hartnack et al. Modelling the many-body dynamics of heavy-ion collisions: Present status and future perspective. *Eur. Phys. J.*, A1:151–169, 1998.

[52] B. Blättel, V. Koch, and U. Mosel. Transport theoretical analysis of relativistic heavy-ion collisions. *Rept. Prog. Phys.*, 56:1–62, 1993.

[53] C. Hartnack and J. Aichelin. Analysis of kaon production around the threshold. *J. Phys. G*, 28:1649–1656, 2002.

[54] P. Crochet et al. Sideward flow of K^+ mesons in Ru+Ru and Ni+Ni reactions near threshold. *Phys. Lett.*, B486:6–12, 2000.

[55] K. Wisniewski et al. Direct comparison of phase space distributions of K^- and K^+ mesons in heavy ion collisions at SIS energies: Evidence for in-medium modifications of kaons? *Eur. Phys. J.*, A9:515–519, 2000.

[56] A. Förster et al. First evidence for different freeze-out conditions for kaons and antikaons observed in heavy-ion collisions. *Phys. Rev. Lett.*, 91:152301, 2003.

[57] J. Cleymans, H. Oeschler, and K. Redlich. Influence of impact parameter on thermal description of relativistic heavy ion collisions at (1–2) AGeV. *Phys. Rev.*, C59:1–11, 1999.

[58] J. Cleymans, H. Oeschler, and K. Redlich. Statistical Model Description of K^+ and K^- Production between 1–10 AGeV. *Phys. Lett.*, B485:27–31, 2000.

[59] S.E. Eiseman et al. Ξ^- production in heavy-ion collisions at the AGS. *Phys. Lett.*, B325:322–326, 1994.

[60] P. Chung et al. Near-threshold production of the multi-strange Ξ^- hyperon. *Phys. Rev. Lett.*, 91:202301, 2003.

[61] D. Elia et al. Hyperon production in 158 and 40 AGeV/c Pb-Pb and p-Be collisions from the NA57 experiment. *Nucl. Phys.*, A734:57–60, 2004.

[62] A. Gobbi et al. A highly segmented ΔE-time-of-flight wall as forward detector of the 4π-system for charged particles at the SIS/ESR accelerator. *Nucl. Instrum. Meth.*, A324:156–176, 1993.

[63] J. Ritman. The FOPI Detector at SIS/GSI. *Nucl. Phys. Proc. Suppl.*, 44:708–715, 1995.

[64] P. Crochet et al. Onset of nuclear matter expansion in Au+Au collisions. *Nucl. Phys.*, A624:755–772, 1997.

[65] A. Andronic et al. Transition from in-plane to out-of-plane azimuthal enhancement in Au+Au collisions. *Nucl. Phys.*, A679:765–792, 2001.

[66] D. Pelte et al. Charged pions from Ni on Ni collisions between 1 and 2 AGeV. *Z. Phys.*, A359:55–65, 1997.

[67] B. Hong et al. Abundance of Delta Resonances in ^{58}Ni + ^{58}Ni Collisions between 1 and 2 AGeV. *Phys. Lett.*, B407:115–120, 1997.

[68] D. Pelte et al. Charged pion production in Au on Au collisions at 1 AGeV. *Z. Phys.*, A357:215–234, 1997.

[69] A. Mangiarotti et al. Sub-threshold phi meson yield in central ^{58}Ni+^{58}Ni collisions. *Nucl. Phys.*, A714:89–123, 2003.

[70] A. Andronic et al. Differential directed flow in Au+Au collisions. *Phys. Rev.*, C64:041604, 2001.

[71] A. Andronic et al. Directed flow in Au+Au, Xe+CsI and Ni+Ni collisions and the nuclear equation of state. *Phys. Rev.*, C67:034907, 2003.

[72] FOPI Collaboration. Technical Proposal. *GSI Report*, 88-03, 1988.

[73] G.F. Knoll. *Radiation Detection and Measurement.* John Wiley & Sons, 1979.

[74] W.R. Leo. *Techniques for Nuclear and Particle Physics Experiments.* Springer-Verlag, 1987.

[75] Georg Goebels. *Untersuchung der Bose-Einstein-Korrelationen geladener Pionen in relativistischen Schwerionenkollisionen der Systeme Au+Au und Ni+Ni im Energiebereich von 1 AGeV bis 2 AGeV.* PhD thesis, University of Heidelberg, 1995.

[76] Edmund Häfele. *Untersuchung der Produktion geladener Pionen im System Au+Au bei 1 AGeV.* PhD thesis, University of Heidelberg, 1995.

[77] H. Drumm et al. Experience with the Jet-Chamber of the jade Detector at PETRA. *Nucl. Instr. Meth.*, 176:333–344, 1980.

[78] R.D. Heuer and A. Wagner. The OPAL Jet Chamber. *Nucl. Instr. Meth.*, A265:11–19, 1988.

[79] H.M. Fischer et al. The OPAL Jet Chamber. *Nucl. Instr. Meth.*, A283:492–501, 1989.

[80] R. Veenhof. *Garfield - Simulation of gaseous detectors.* CERN, http://consult.cern.ch/writeup/garfield, 2001.

[81] W. Blum and L. Rolandi. *Particle Detection with Drift Chambers.* Springer-Verlag, 1994.

[82] Tarek Kreß. *Elliptischer Fluß von Protonen und leichten Kernen in Au+Au-Reaktionen bei Strahlenergien zwischen 400 und 1490 AMeV.* PhD thesis, Technische Universität Darmstadt, 2002.

[83] Dieter Best. *Tracking mit der Hough-Transformation für die Zentrale Driftkammer des GSI-4π-Experiments.* Diplomarbeit, Universität Mainz, 1992.

[84] Christopher H. Pinkenburg. *Flußeffekte geladener Pionen im System Au+Au bei $E_p = 1$ AGeV.* PhD thesis, University of Heidelberg, 1995.

[85] CES Creative Electronic Systems S.A. *RIO3 8064, A PowerPC-Based VME Real-Time Processor Board.* http://www.ces.ch/.

[86] J. Hoffmann. Steuerungs- und Auslesemodul – Technische Spezifikationen. Technical report, Gesellschaft für Schwerionenforschung mbH, 2002.

[87] J. Friese et al. Studying In-Medium Hadron Properties with HADES. *Prog. Part. Nucl. Phys.*, 42:235–245, 1999.

[88] G. Kraft. Tumor therapy with heavy charged particles. *Prog. Part. Nucl. Phys.*, 45:473–544, 2000.

[89] Texas Instruments. *TMS320C6000 Technical Brief*, February 1999.

[90] Texas Instruments. *TMS320C6000 Peripherals Reference Guide*, April 1999.

[91] P. Cavata et al. Determination of the impact parameter in relativistic nucleus-nucleus collisions. *Phys. Rev.*, C42:1760–1763, 1990.

[92] P. Danielewicz and G. Odyniec. Transverse momentum analysis of collective motion in relativistic nuclear collisions. *Phys. Lett.*, B157:146–150, 1985.

[93] Ralf Kutsche. *Untersuchungen der In-Medium-Eigenschaften von K^0_S-Mesonen und Λ-Hyperonen an der Produktionsschwelle.* PhD thesis, Technische Universität Darmstadt, 2000.

[94] J.Y. Ollitrault. Flow systematics from SIS to SPS energies. *Nucl.Phys.*, A638:195–206, 1998.

[95] I.N. Bronstein and K.A. Semendjajew. *Taschenbuch der Mathematik.* B.G. Teubner Verlagsgesellschaft, Verlag Nauka Moskau, 1991.

[96] J.Y. Ollitrault. Reconstructing azimuthal distributions in nucleus-nucleus collisions. *nucl-ex/9711003*, 1997.

[97] K. Hagiwara et al. Review of particle physics. *Phys. Rev.*, D66:010001+, 2002.

[98] D. Drijard, H.G. Fischer, and T. Nakada. Study of event mixing and its application to the extraction of resonance signals. *Nucl. Instrum. Meth.*, 225:367–377, 1984.

[99] D. L'Hôte. About resonance signal extraction from multiparticle data: combinatorics and event mixing methods. *Nucl. Instrum. Meth.*, A337:544–556, 1994.

[100] GEANT Team. *GEANT - Detector Description and Simulation Tool.* CERN, http://consult.cern.ch/writeup/geant, 1995.

[101] P.J. Siemens and J.O. Rasmussen. Evidence for a Blast Wave from Compressed Nuclear Matter. *Phys. Rev. Lett.*, 42:880–883, 1979.

[102] B. Hong et al. Stopping and Radial Flow in Central ^{58}Ni + ^{58}Ni Collisions between 1 and 2 AGeV. *Phys. Rev.*, C57:244–253, 1998.

[103] G. Poggi et al. Evidence for collective expansion in light-particle emission following Au+Au collisions at 100, 150 and 250 *A*MeV. *Nucl. Phys.*, A586:755–776, 1995.

[104] J.L. Ritman, N. Herrmann, D. Best, and the FOPI collaboration. On the transverse momentum distribution of strange hadrons produced in relativistic heavy-ion collisions. *Z. Phys.*, A352:355–357, 1995.

[105] M. Justice et al. Λ hyperons in 2 AGeV Ni+Cu collisions. *Phys. Lett.*, B440:12–19, 1998.

[106] P. Chung et al. Neutral strange particle production and flow at AGS energies. *J. Phys. G*, 25:255–261, 1999.

[107] C. Pinkenburg et al. Production and collective behavior of strange particles in Au+Au collisions at 2-8 AGeV. *Nucl. Phys.*, A698:495–498, 2002.

[108] P. Chung et al. Anti-flow of K^0_S Mesons in 6 AGeV Au + Au Collisions. *Phys. Rev. Lett.*, 85:940–943, 2000.

[109] L.A. Winckelmann. Microscopic calculations of stopping, flow and electromagnetic radiation from 160AMeV to 160AGeV. *Nucl. Phys.*, A610:116–123, 1996.

[110] G.Q. Li and C.M. Ko. Λ flow in heavy-ion collisions: The role of final-state interactions. *Phys. Rev.*, C54:1897–1902, 1996.

[111] G.F. Bertsch and S. Das Gupta. A Guide to microscopic Models for intermediate Energy Heavy-Ion Collisions. *Phys. Rept.*, 160:189–233, 1988.

[112] J. Aichelin. "Quantum" Molecular Dynamics – A dynamical microscopic n-Body Approach to investigate Fragment Formation and the nuclear Equation of State in Heavy-Ion Collisions. *Phys. Rept.*, 202:233–360, 1991.

[113] P. Braun-Munzinger, J. Stachel, and C. Wetterich. Chemical Freeze-out and the QCD Phase Transition Transition. *Phys. Lett.*, B596:61–69, 2004.

[114] J. Cleymans, D. Elliott, A. Keränen, and E. Suhonen. Thermal model analysis of particle ratios in Ni+Ni experiments using exact strangeness conservation. *Phys. Rev.*, C57:3319–3323, 1998.

[115] C.-Y. Wong. *Introduction to High-Energy Heavy-Ion Collisions.* World Scientific, 1994.

Danksagung

Viele Beteiligte haben einen Beitrag zum Gelingen dieser Arbeit geleistet. Hier möchte ich mich nun bei ihnen bedanken.

An erster Stelle danke ich Prof. Dr. Norbert Herrmann für die freundliche Aufnahme in die FOPI-Familie und für die hervorragende Betreuung dieser Arbeit. Durch viele Diskussionen mit ihm habe ich einen guten Einblick in das Gebiet der Schwerionenphysik erhalten.

Für viele konstruktive Beiträge möchte ich Dr. Alessio Mangiarotti, Dr. Andreas Schüttauf und ganz besonders Dr. Yvonne Leifels danken, die mir auch beim Bändingen der Bits und Bytes für die Datenreduktion eine große Hilfe war. Ein weiteres Dankeschön geht an Cornelia Sander für Rat und Tat in sprachlichen Dingen.

Mein Dank für die gute Zusammenarbeit gilt natürlich auch den weiteren Mitgliedern der Heidelberger Gruppe: Mohamed Lotfi Benabderrahmane (Thanks also for your Couscous), Everard Cordier und Prof. Dr. Dietrich Pelte sowie meinen ehemaligen Zimmerkollegen Dr. Marc Stockmeier, Dr. Krzyzstof Wiśniewski und Thomas Lehmann, der als Gast in unserem Büro weilte.

Je tiens a remercier Xavier Lopez, Dr. Philippe Crochet et Dr. Nicole Bastid de LPC Clermont-Ferrand pour les échanges d'idées que j'avais eu avec eux, qui ont contribuées énormément à l'analyse des données. Je les remercie aussi pour leurs hospitalité au cours de mon séjour à Clermont-Ferrand.

Auch der FOPI-Gruppe an der GSI möchte ich hier meinen Dank aussprechen: Dr. Klaus Hildenbrand für die Erläuterung technischer Details des Detektors, Dr. Piotr Koczon für Hilfe bei der Datenaufnahme und Dr. Willibrord Reisdorf für Anmerkungen zur Datenanalyse. Weiterhin gilt mein Dank Dr. Anton Andronic, Dr. Olaf Hartmann, Dr. Marek Kirejczyk, Dr. Tarek Kress, Dr. Young-Jin Kim, Victor Simion, Zbigniew Tyminski, Dr. Zhigang Xiao, Igor Yushmanov und Dr. Alexander Zhilin. Nicht zu vergessen sind an dieser Stelle auch die Sekretärinnen Aloisia Busch und Getraude Eisold, sowie die Techniker Gert Augustinski, Michael Marquardt und Claus Schröder, die sich fürsorglich um das Magnetfeld gekümmert haben.

Ein weiteres Dankeschön geht an die Kollaborateure Dr. Byungsig Hong, Dr. Joszef Kecskemeti, Dr. Milorad Korolija, Dr. Roland Kotte, Dr. Mihai Petrovici, Dr. Fouad Rami, Dr. Brunon Sikora und viele weitere die dazu beigetragen haben, das Ξ-Experiment erfolgreich durchzuführen.

Ebenso bedanke ich mich bei Prof. Dr. Johanna Stachel für die Anfertigung des Zweitgutachtens sowie bei Prof. Dr. Hans-Jürgen Pirner und Prof. Dr. Volker Lindenstruth für ihre Bereitschaft, die Prüfungsbereiche Theoretische Physik und Technische Informatik zu übernehmen.

Ganz zum Schluss, aber auch ganz besonders, danke ich meinen Eltern und meiner Schwester für ihre Unterstützung auf die ich mich immer verlassen konnte.