JTAG Boundary-Scan of the Belle II Pixel Vertex Detector

JTAG Boundary-Scan des Belle II Pixel Vertex Detektors

Masterarbeit

Fakultät für Physik
Ludwig-Maximilians-Universität
München

eingereicht von: Philipp Leitl

eingereicht am: 01. Dezember 2015

Betreuer: Prof. Dr. Christian Kiesling
Dr. Hans-Günther Moser

Abstract

The High Energy Accelerator Research Organization (KEK) is operating Japans largest particle physics laboratory. The particle accelerator KEKB, which is located in Tsukuba, collides electrons and positrons with a center of mass energy of about 11 GeV. These collisions produce a lot of different new particles. Some of them are able to leave the beam pipe, others decay before they even reach it. Two of these short living particles are of special interest: the neutral B_d^0 meson and its anti-particle, the $\bar{B}_d^0$ meson.

A different behavior of particle and anti-particle breaks the CP-symmetry. The understanding of this CP violation may help explain the origin of the universe, where matter and anti-matter were created in equal parts, but after a process of annihilation the today visible universe remained as surplus.

It was the objective of the Belle experiment to study the influence of CP violation on this observed asymmetry between matter and antimatter. For this purpose the collision products of KEKB were recorded by a detector system also called Belle. During its runtime from 1999 to 2010 the experiment provided first proof of CP-violation in the B meson system, measured a lot of decay channels of these particles and observed a number of new particles. The BaBar experiment at SLAC was built for the same purpose and could confirm the Belle data. After the successful verification through the experimental results, Makoto Kobayashi and Toshihide Maskawa received the 2008 Nobel Prize in Physics for their work on CP violation.

To continue the research on this field the accelerator at KEK is at the moment upgraded to SuperKEKB to reach a new design luminosity of $8 \times 10^{35}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$. This means, that the upgraded accelerator will have a much higher collision rate. In combination with the aim to make measurements with even more precision, it became necessary to upgrade the detector as well.

The new Belle II detector is a combination of improvements of the Belle detector and the introduction of new technologies. One of the new parts will be the innermost subdetector, the Pixel Vertex Detector, which will help to measure decay vertices with a resolution of about 20 µm. This detector is currently under construction at the Max Planck Institute for Physics in Munich in collaboration with several institutes all over Europe.

The sensors are produced at the semiconductor laboratory of the Max Planck Institute in Munich. The detector consists in total of almost 8 million DEPFET pixels. This new technology requires specific electronics for controlling and readout. Therefore three different kinds of ASICs are mounted very close to the sensors. Because of space limitations ball grid arrays are used for the contacts to the electronic circuitry.

For placing the ASICs onto the sensor module a so called flip chip method is used. After the mounting process a quality assurance is needed to verify that the integration was done correctly and that the circuitry and the electronics are working properly. The designers and

the production crew decided to use boundary-scan tests, an industrial standard, as part of the quality assurance.

The task of the author was to prepare a testing setup and to implement boundary-scan in time for the series production of the sensor modules. It was necessary to prepare the basic conditions in terms of hardware and software.

To gain access to the required nets a breakout board was designed and produced. The description of the boundary-scan implementation in the ASICs, so called BSDL files, were missing or not complete. Also the netlist file, which describes the circuitry, had to be adapted to match the different versions of sensor modules and the ASIC description in the BSDL files. These files were created following the guidelines of the relevant IEEE standard and the special requirements of the acquired commercial boundary-scan software.

After the completion of the single parts, the combined system was tested and further optimized. The available preproduction modules of the senor were used for the first measurements. From these results a dedicated boundary-scan project file was created for the series production modules. Also a test setup for the quality assurance during the series production was prepared including a probe station with needle cards.

For the final design of the sensor modules no changes are expected, which would have influence on the performance of the boundary-scan system. In this way it is now capable to contribute its part to the quality assurance process during the production of the sensor modules for the Pixel Vertex Detector.

Contents

List of Figures

Part I.

Physical Motivation and Detector Information

1

Introduction

According to the current knowledge, time and room started to exists with the “big bang” about 13.8 billion years ago. A big amount of energy was released and the universe consisted of a very hot and dense quark-gluon plasma. Particle and anti-particle pairs were continuously created and destroyed. A little fraction of these processes favored matter over anti-matter. The so called baryogenesis resulted in a universe which consists almost absolutely out of matter. A lot of effort is made to examine the processes in the early universe to understand the differences between matter and anti-matter.

The universe has cooled down and its early state must artificially be reproduced. Huge machines accelerate particles to almost the speed of light and collide them. The local energy density is high enough to simulate the conditions after the “big bang”. New, heavier particles are produced, which decay into lighter ones, as the first heavy particles did 13.8 billion years ago. With a precise measurement of these decays, it is possible to learn more about the nature of matter and anti-matter.

1.1. Standard Model of Particle Physics and Symmetries

The present picture about the nature of matter and anti-matter is described by the Standard Model of particle physics (SM). It is a theory which describes the electromagnetic, weak and strong interactions between the elementary particles. The discovery of the Higgs Boson completed the set of particles which are required by the SM [1].

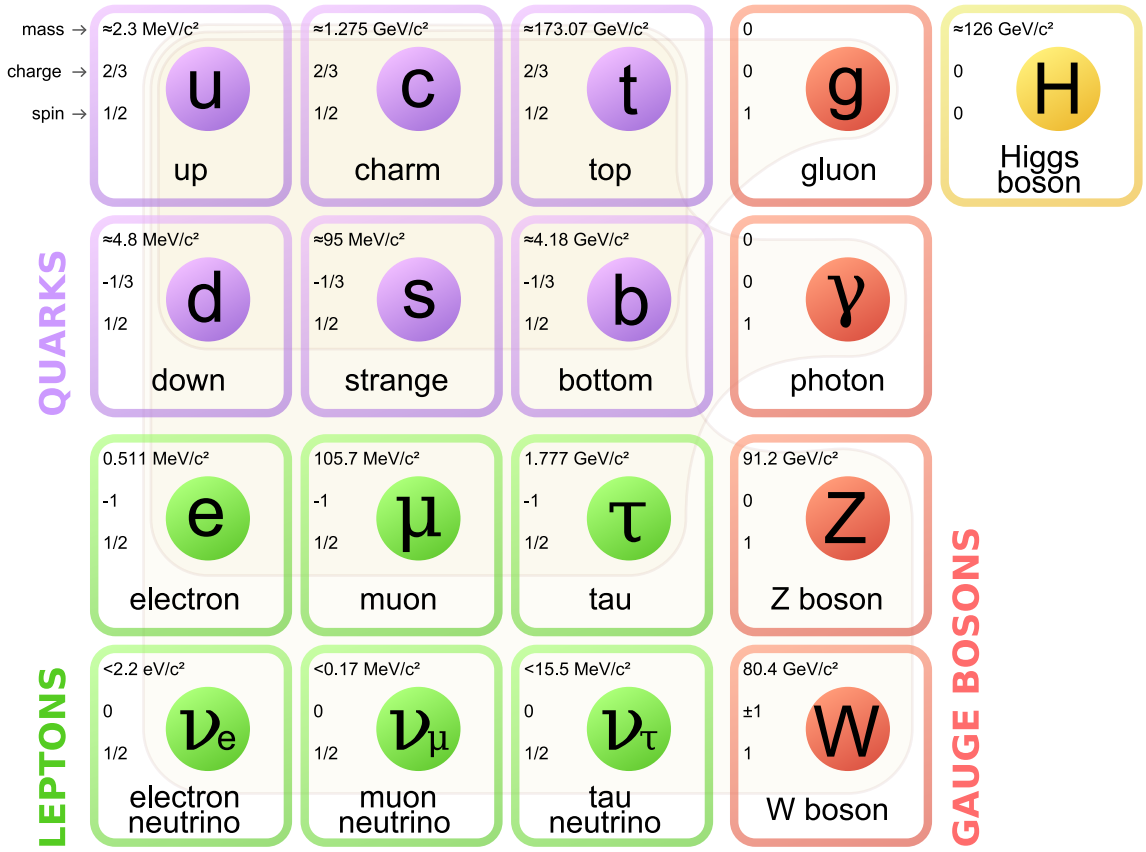


Figure 1.1.: Elementary particles and gauge bosons [2]

The SM is very successful in describing the kinematics and interactions of elementary particles. Several particles were precisely predicted and found to have the expected properties. All particle physics experiments confirmed the SM with very high accuracy. However, the SM does not integrate gravity and fails in describing several observed phenomena. The accelerating expansion of the universe is likely caused by dark energy and dark matter, which have no equivalent in the SM. The observed baryon asymmetry is much bigger than described in the SM as well.

Symmetries play an important role in the SM. The Noether Theory correlates fundamental symmetries and conservation laws [3]. Important discrete symmetries are Parity (P), Charge conjugation (C) and Time reversal (T). As they are correlated to conservation laws, they were considered to be obeyed by every particle interaction. However, the weak interaction was discovered to violate the parity maximal. The combination of C- and P-

symmetry seemed to rescue the conservation laws, but it turned out that a violation of the CP symmetry exists as well. The different behavior of matter and anti-matter was introduced into the SM by Kobayashi and Maskawa. The quark mixing described in the CKM matrix accounts for CP violation. However, is effect is too small and physics beyond the SM is searched to answer the remaining questions.

1.2. CP Violation in Decays of B Mesons

The first hint of CP violation was found in the neutral Kaon system. Cronin and Fitch found in 1964 that K_L mesons can mix with their short living counterpart, K_S mesons. After "filtering" a beam of K_L mesons, they observed decay particles which came from K_S mesons. The effect is very small in relatively common decays [3].

Neutral B mesons can mix as well. For them the CP violation is a large effect in extremely rare decays. The lifetime of 1 ps for a B meson is relatively long compared to 1 fs for the D mesons. It is easier to study the decays of B mesons. Therefore, dedicated "B-factories" were built at SLAC and KEK to produce an enormous amount of neutral B mesons. The experiments BaBar and Belle, respectively, succeeded to measure CP violation in the system of neutral B mesons [3].

1.3. Electron-Positron Collider SuperKEKB

The SuperKEKB is a circular electron-positron collider at the High Energy Accelerator Research Organization (KEK) in Tsukuba, Japan. It is an upgrade of the KEKB machine and will reach a luminosity of $8 \times 10^{35}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$. To achieve this number, the beam currents will be doubled in comparison to KEKB. This is combined with the use of a so called "nano-beam" scheme. The asymmetry in the beam energies will be decreased to 7 GeV for the electron beam and 4 GeV for the positron beam, resulting in a Lorentz boost of $\beta\gamma = 0.28$ for the produced particles [4]. The center of mass energy corresponds to the $\Upsilon(4S)$ resonance ($=10.58\,\mathrm{GeV}/c^2$), which is slightly above the threshold of B meson pair production. The production cross section is roughly 1 nb [4].

The lifetime of the B mesons is not long enough that they could be detected directly. They decay before they reach the beam pipe. Their decay products, however, are detected and their vertices reconstructed.

1.4. Belle II Detector

The Belle II detector system is placed around the interaction point. It consists of several sub-detectors, which have special purposes.

The K_L and muon detector (KLM) is the outermost part. It literally shields the detector with massive iron plates, which are used as stopping material for neutral kaons and muons. These particles create showers of ionizing particles in the iron. The showers are detected within glass electrode resistive plate chambers, which are sandwich like stacked with the iron plates.

Inside of the KLM, a superconducting solenoid provides a magnetic field of 1.5 T. The curvature of charged particles crossing the magnetic field is used to calculate their momenta.

The next sub-detector is the electromagnetic calorimeter (ECL). It is mainly used for precise measurements of photon energies and positions. It consists of thallium-doped CsI crystals, which emit light when they are hit by a particle. The ELC is also used for trigger generation and luminosity measurements.

The ELC encloses the particle identification device (PID). Particles crossing its quartz radiators emit Cherenkov photons. By measuring the propagation time of the emitted photons, a reliable decision between kaons and pions is possible. For the forward endcap a special Aerogel Ring-Imaging Cherenkov detector (ARICH) was designed for this task.

The Central Drift Chamber (CDC) is a gaseous detector filled with helium and ethane. More than 14000 sense wires cross its volume and provide precise measurement of particle tracks as well as momenta. The CDC is able to identify low momenta particles, which do not reach the outer detectors, by measurements of the energy loss. It also provides trigger signals for charged particles.

The next sub-detector is the vertex detector. It consists of four layers with double-sided silicon strip sensors (SVD) and two layers with silicon pixel sensors (PXD). It was necessary to use a pixel detector for the two innermost layers because of the increase in radiation due to the higher luminosity and a reduction of the beam pipe radius.

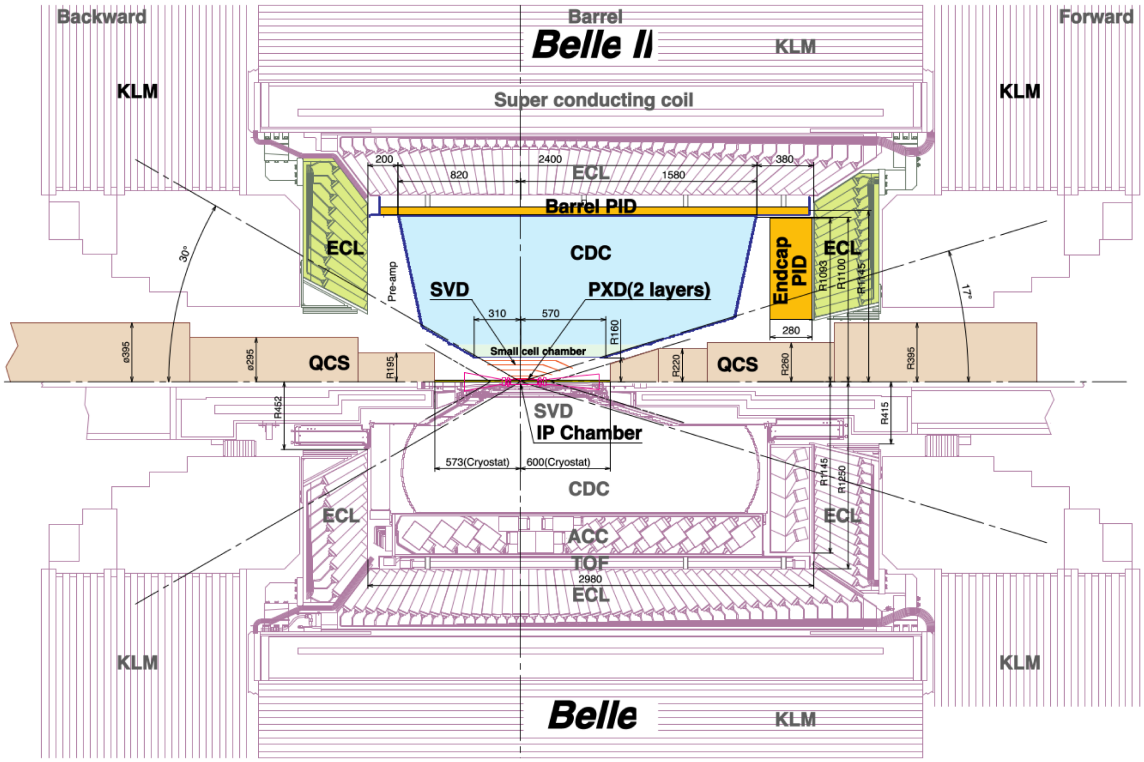


Figure 1.2.: Belle II detector system (top half) compared to the Belle detector (bottom half).

The Belle II PXD will be one of the most advanced pixel detectors ever developed for a particle physics experiment.

Belle II Technical Design Report

2

Pixel Vertex Detector

The inner most part of the Belle II detector is the Pixel VerteX Detector (PXD). Its sensitive area consists of an arrangement of pixels as in a LCD screen. The purpose of the PXD is to detect traversing particles with a very high spacial accuracy, so that it becomes possible to calculate where the particles came from. The point in space, where a particle decays, and where the decay products emerge from, is called vertex. Also the energy and the charge of the particles are of interest to identify them and to study the processes at the interaction point. Although, the PXD is also able to perform energy measurements in a certain range, other sub-detectors specialize on these issues.

Due to their short lifetime, the B mesons decay within the beam pipe and can't be detected directly. Just their decay products leave the beam pipe and cross the detector, where their hits are recorded to calculate the vertices. Possible candidates are K and π mesons and all leptons except the neutrinos, which interact weakly. The main goal of the Belle II experiment is to study the CP violation in the decay of B mesons. The PXD has to meet specific requirements to make these studies possible. The mean distance between a B and a $\bar{\mathrm{B}}$ vertex is $\Delta z \approx 150\,\mu\mathrm{m}$. It is necessary to definitely distinguish between these vertices and a spatial resolution of less than $20\,\mu\mathrm{m}$ is needed [5].

To reconstruct a trajectory of a traversing particle at least two points in space are mandatory, assuming that the flight of the particle was straight in this section. If the energy of the particle is low and its trajectory therefore significantly bending or curling in the magnetic field of the detector, more points are needed. In Belle II the system of CDC, SVD and PXD together measure these points of particle trajectories. The more points are available the better the reconstruction gets and with a higher precision of position measurements the resolution of the vertex increases [5]. The PXD is surrounded by four layers of the SVD and consists itself of two layers of DEpleted P-channel Field Effect Transistor (DEPFET)

pixel sensors. The radius of the inner and outer layer are 14 mm and 21 mm respectively. The pixels have different sizes of (55, 60, 70, 85) $\times$ 50 μm^2, where the smaller ones are on the inner layer and closer to the interaction point. The inner layer consists of eight and the outer one of twelve ladders arranged cylindrically and overlapping around the beam pipe to cover the entire acceptance of the tracking system around the interaction point (Figure 2.1). The acceptance in the azimuth angle is between 17° and 150° [4]. This asymmetry accommodates to the asymmetric beam energies. The particles have a boost in z direction and the acceptance in forward direction is therefore higher.

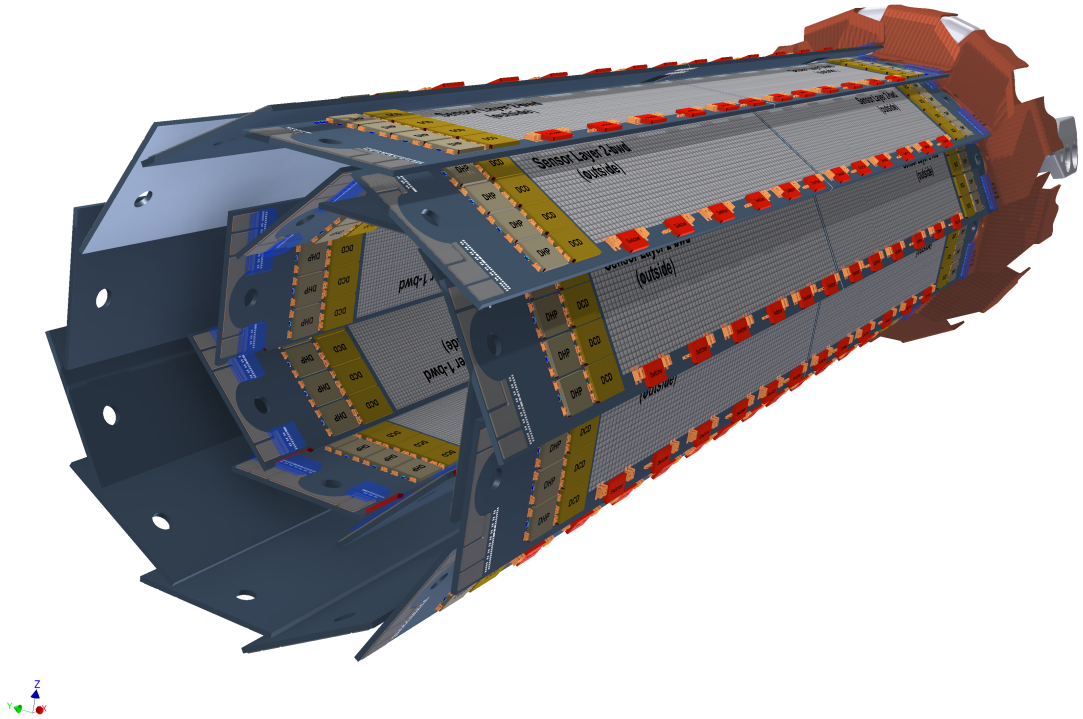


Figure 2.1.: Pixel Vertex Detector

One big development of the accelerator SuperKEKB is an increase in luminosity by a factor of 40 compared to the predecessor KEKB accelerator. This leads to a much higher rate of interesting reactions, but in the same way the disturbing background increases. The higher luminosity results in a higher occupancy of the detector, which makes the usage of a pixel detector absolutely necessary [4]. Too many hits at one time on a strip detector causes ambiguity and the hits can't be allocated clearly. Nevertheless, the readout time of a pixel detector should also be very fast, so that tracks can be assigned to a common event.

2.1. Depleted P-channel Field Effect Transistor Sensors

The PXD uses DEPFET sensors for its sensitive area. The decision for this technology was done for the fact, that the sensors can be produced very thin. Other pixel sensors, which are successfully used in particle physics experiments as for e.g. ATLAS, are much thicker. This does not represent a problem when the momenta of the particles are high in comparison to the Belle II experiment, where the energies of the decay products are in the range of several 100 MeV. At these energies there is a high possibility that a charged particle interacts with an electron of the detector material and gets deflected. A deflected particle, however, loses its vertex information. Therefore, the vertex resolution at Belle II is dominated by Multiple Coulomb scattering and the material budget in the acceptance area must be kept as low as possible [6].

The DEPFET technology was first proposed in 1987 by J. Kemmer and G. Lutz and experimentally confirmed three years later [7, 8]. In addition to the possibility to produce very thin detectors the DEPFET technology has some further advantages. It can be produced using an inherently radiation-hard technology, can be operated consuming very little power and has an excellent signal to noise ratio [4, 6]. Furthermore, the design of a DEPFET pixel can be adjusted for special needs, so that there are rectangular or circular designs, linear or more complex signal amplifications. Its not surprising that there is a big interest to use this technology for particle detectors not only for Belle II, but also at the ILC, XFEL and ATHENA+ x-ray observatory for example [9]. The first operating linear DEPFET detector will be the PXD.

2.2. Working Principle of a DEPFET Pixel

A sketch of a DEPFET pixel as it is used for the PXD is shown in figure 2.2. It consists of a n-doped silicon bulk with a p^+ implant at the backside. By applying an appropriate voltage to the p^+ implant, the silicon bulk gets fully depleted and becomes a silicon drift chamber. Therefore, it represents the radiation sensitive part of the pixel. On the top a p-channel Metal-Oxide-Semiconductor Field Effect Transistor (MOSFET) is processed, where the voltage at the gate contact controls the conductivity between source and drain.

If an ionizing particle passes through a DEPFET pixel, it interacts with the electrons of the silicon and creates electron-hole pairs in the silicon bulk. Due to the depletion, the holes are attracted to the p^+ implant at the backside, whereas the electrons are collected in a potential minimum, which is placed $\sim$1 µm underneath the external gate of the MOSFET. The collected electrons influence the conductivity of the p-channel of the MOSFET by inducing image charges in it. Because of this capacitive coupling, the spot where the electrons are collected is called internal gate [10]. Particles with a higher potential to ionize create more electrons and the conductivity of the MOSFET increases linearly. The current indicates if a particle crossed the DEPFET and how big its ability to ionize was.

This technology provides an internal amplification of the signal, because the current is increased by ∼400 pA per signal electron in the internal gate [11]. A readout of a pixel has no influence on the signal electrons as they remain in the internal gate. A repeated measurement of the same signal is hence possible. However, in order to reset the pixel and to get rid of thermally induced charges an additional clear contact is needed. By applying a sufficiently high positive voltage to this clear contact, the potential minimum for the free electrons is no longer the internal gate but the clear contact. They drift out of the DEPFET and the pixel is sensitive again [11, 10].

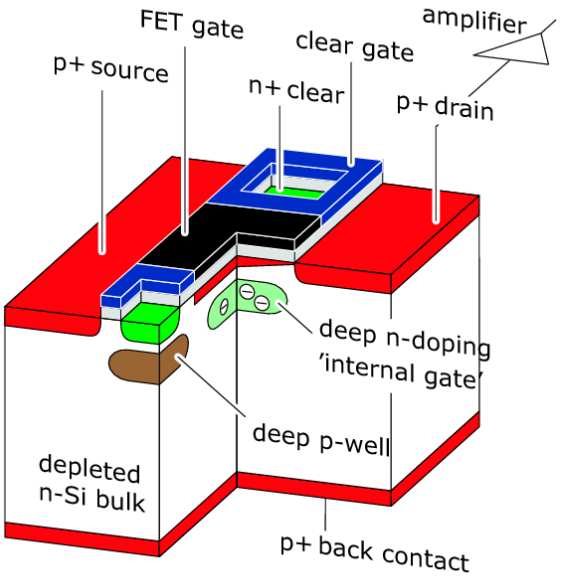


Figure 2.2.: Schematic of a Depleted Field Effect Transistor [4]

It is possible to use the clear mechanism in a suppressed way, so that new created electrons from the silicon bulk drift directly to the clear contact, the previous collected electrons in the internal gate however, stay there. This so called gated mode allows to set the DEPFET pixels insensitive for a defined time interval. Afterwards the saved signal can be read out as usual. This unique feature will be used to protect the PXD from saturation due to so called "noisy bunches", which occur after the injection of new particles into the accelerator rings [12].

Another aspect in comparison to Charge-Coupled Device (CCD) is that the collected signal electrons are not moved to a readout device, which can cause signal losses. Instead every signal is read out at the very pixel where it occurred. The electrons are also not used as input for a charge amplifier, where they get lost. In a CCD only one readout is possible. To exploit the advantages of the DEPFET technology, a totally different readout system has to be used. The next sections give an overview, how this is realized for the PXD.

2.3. Readout of the DEPFET sensors

To readout the information stored in the pixels Application-Specific Integrated Circuits (ASICs) are used to perform a lot of complex activities. The basics of a simple readout and clear procedure as it is used for the PXD modules is described below.

The signal electrons accumulate in the internal gate where they have influence on the current of the MOSFET. Therefore, it is necessary to know, what the standard current is, when the external gate is switched on, but there are no electrons in the internal gate. This value is called offset or pedestal current. The difference between the pedestal current and the measured current is the signal, which indicates, if the pixel was hit by an ionizing particle.

To get the two values, two readouts are required. This can be performed in two different ways. One way would be a first readout followed by a clearing of the internal gate and then a second readout with the empty gate. After that, both values are compared. With this procedure, two readouts for every measurement are necessary. The other way would be to measure the pedestal value at first and to save it. All following measurements will then be compared to the stored pedestal value. This can reduce the readout time almost by a factor of two. However, the system has to take care for the storage of the pedestal values. One has also to consider, that the pedestal values can change under the influence of radiation, temperature or other effects [13].

The sensitive area of one module consists therefore of a matrix of 250×768 pixels [11]. There are two possible extremes for a readout of a DEPFET pixel matrix. The fastest way is to connect each pixel to an own readout electronic. A complete frame (readout of all pixels) could then be taken at once. This method needs a lot of space for the individual connections from the pixels to their own readout electronics. The other extreme is to use just one readout device. The pixels are read out one after another and in between, the lines must be switched, so that the next pixel is connected to the readout device. For this method the readout time explodes [13].

A practical way is a compromise between these extremes. For the PXD it is taken advantage of the fact, that a charge collection in the internal gate can influence the DEPFET current when the external gate is switched on, but it is not sufficient to activate a DEPFET, when the external gate is off. In this way the DEPFET pixels are still sensitive to ionizing particles at all times and collect the signal charge, but they are only activated for readout, when the external gate is switched on [13].

For the PXD modules a four-fold readout system is applied, where always four rows are read out at one time. The routing concept with the electrical connections of the DEPFET matrices is shown in Figure 2.3. Four following rows are connected to one gate and one clear line, which are controlled by an ASIC called Switcher. When a set of four lines is activated, in total 1000 pixels are active and ready for readout. Therefore, 1000 devices are necessary to

convert the drain current to a digital value [1]. These Analog-to-Digital-Converters (ADCs) are provided by 4 readout ASICs, the Drain Current Digitizers (DCDs), on each module. After the readout of the four lines, a clear pulse is applied by the Switcher and the procedure begins with the next four rows. A single readout of one DEPFET pixel lasts about 100 ns. The readout time for four lines is the same, because they are read out in parallel. For all pixels of a module it takes approximately $768/4 \cdot 100\,\text{ns} \approx 20\,\mu\text{s}$. This is the time for the whole PXD, because all the modules are read out in parallel.

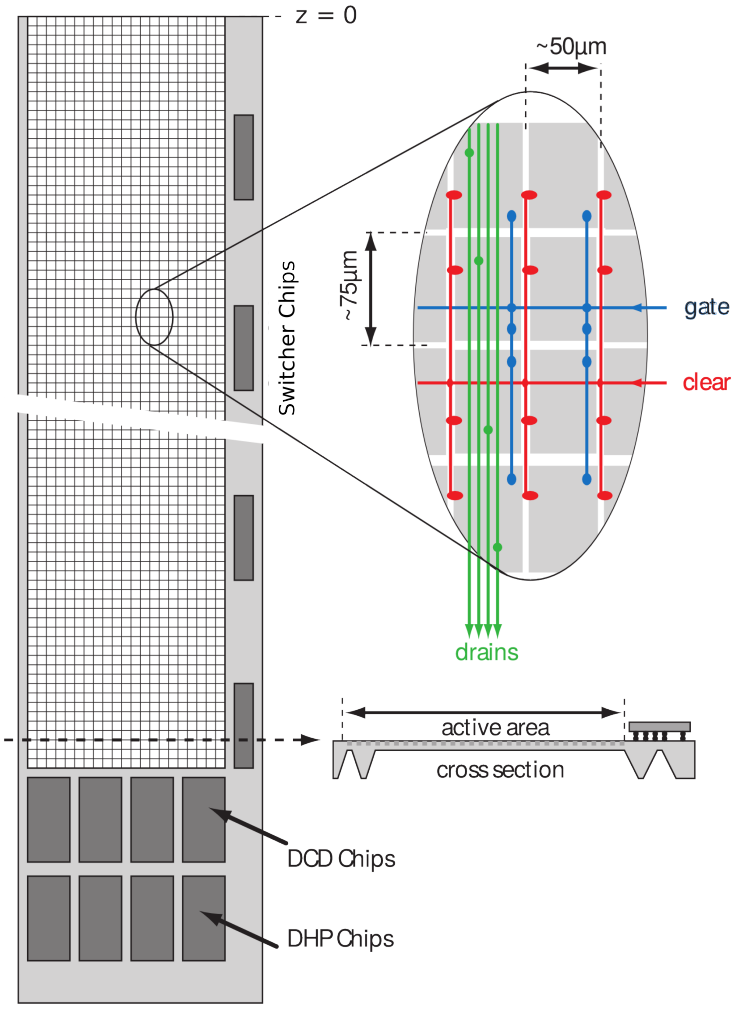


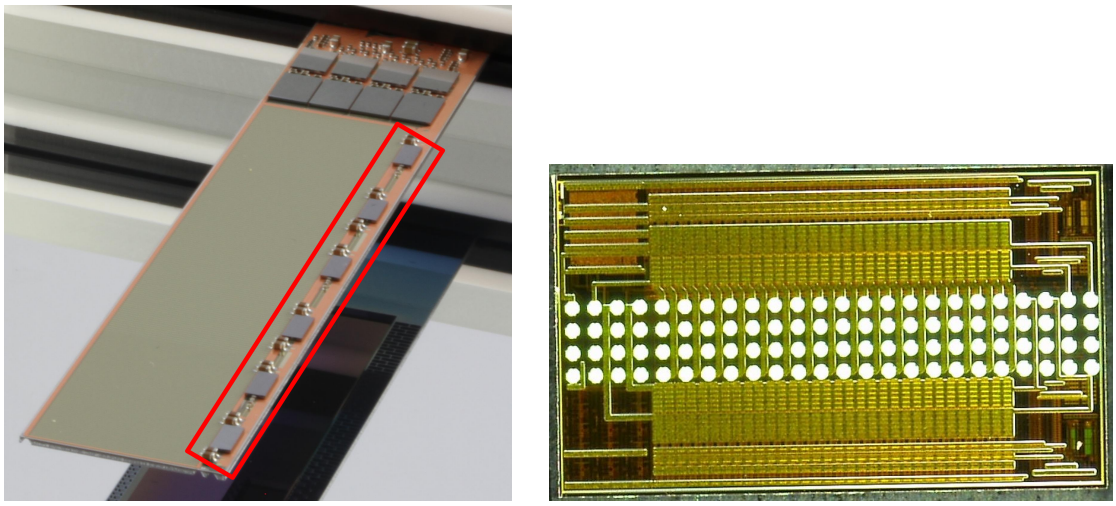
Figure 2.3.: Routing concept of the DEPFET matrix on a PXD module [4].

[1] There are two different ways for the readout: the Source Follower and the Drain Readout. The latter is the faster one. Hence, the Drain Readout is used [13].

2.4. Switcher

The Switcher controls the gate and clear lines of the DEPFET matrix. Six Switchers placed on the so called "balcony", the long side of the matrix on each module. Each Switcher has 32 gate and clear lines and controls therefore 128 matrix rows. The information when and how to switch on the matrix comes from the Data Handling Processor (DHP) and runs through the Switchers in a shift register, so that they are always working one after another.

In Figure 2.4 (b) a photomicrograph of the Switcher is shown. The 96 bright, circular spots are the ASIC pins, the contacts to the chips electronic. They are interconnected via flip chip to the module. The size of the Switcher is about $2 \times 3.6\,\mathrm{mm}^2$ [4].

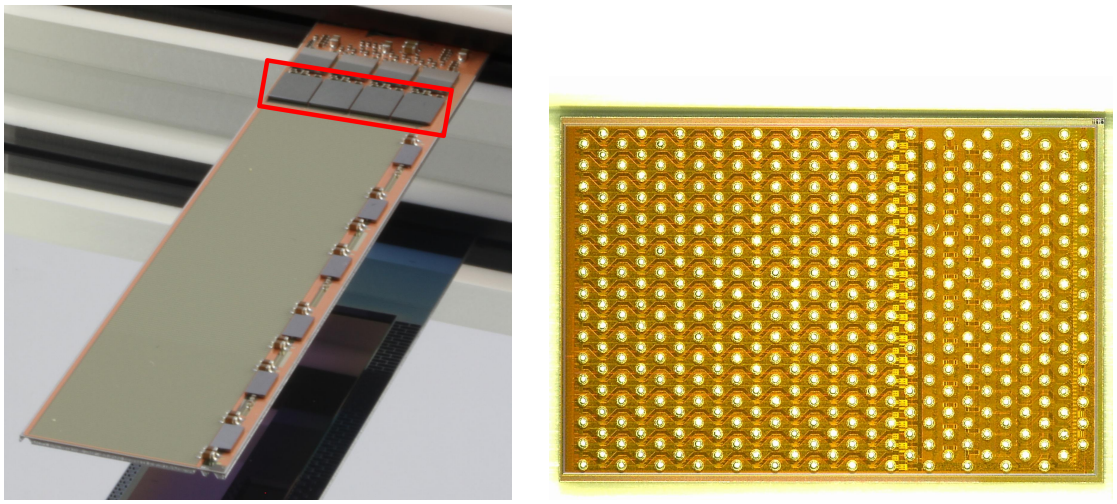


(a) **(b)**

Figure 2.4.: In Figure (a) the position of the 6 Switchers on a PXD module is shown. They are placed on the balcony along the long side of the matrix. The second Figure (b) shows a photomicrograph of the top side of a Switcher.

2.5. Drain Current Digitizer

The task of the DCD is the digitization of the drain currents from the DEPFET pixels. Every chip has therefore 256 analog input channels, 250 of which are connected to the matrix. Four DCDs are mounted to the module, therefore, 1000 drain lines can be read out at the same time. After the conversion, the digital data is transferred to the DHP chips using fast parallel 8-bit digital outputs. One DCD has 431 pins and an approximate size of $3.2 \times 5\,\mathrm{mm}^2$. The left side of the Figure 2.5 (b) shows the analog part of a DCD, which is orientated to the matrix, the right side shows the digital part, which is connected to a DHP [4].



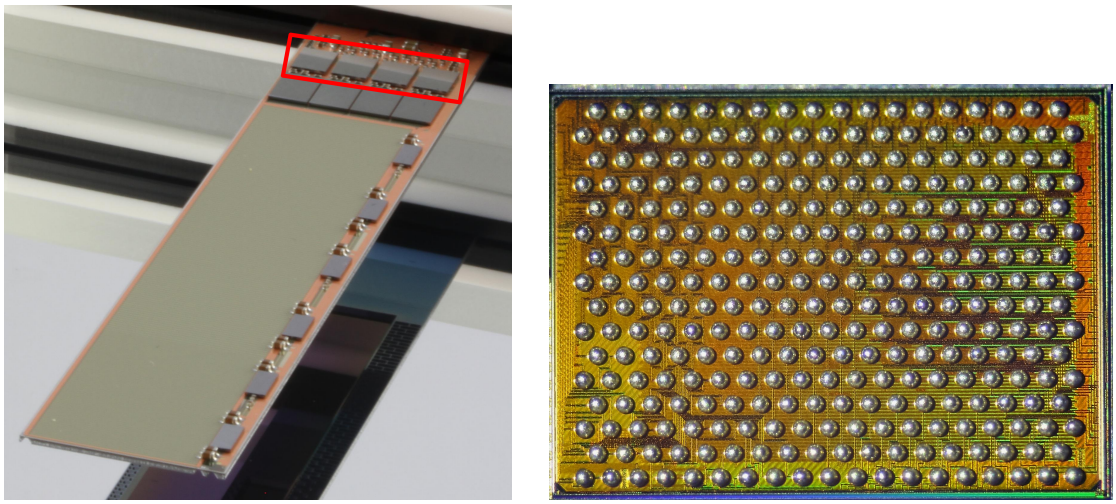
(a) (b)

Figure 2.5.: On Figure (a) the position of the 4 DCDs on a PXD module is shown. They are placed on the end-of-stave along the short side of the matrix. The second Figure (b) shows a photomicrograph of the top side of a DCD.

2.6. Data Handling Processor

The DHP receives the data from the DCD and processes them to reduce the data rate. This is achieved by zero-suppressing the raw data and reading out only triggered events. Zero-suppressing means that only pixel values which show a significant higher current compared to their pedestal value are stored. In order to do this, the DHP compensates also for common mode noise and pedestal fluctuations. A common mode noise affects all values of a sample. The DHP calculates therefore the mean signal value and subtracts it from the raw data. The processed data sample and a time stamp are stored in a buffer until a trigger signal from the Data Handling Engine (DHE) requests them. In this case the stored data is transmitted by Low Voltage Differential Signaling (LVDS) high speed lines, which achieve a data rate of 1.25 Gbit/s per chip, resulting in a total of 5 Gbit/s per module.

The DHP has also the task to control and synchronize the activities on the module. It provides the clock signal for the fast processes as well as the Joint Test Action Group (JTAG) interface for configuration and slow control. One DHP has 296 pins, but 41 are vacant, resulting in 255 active pins. The size of the chip is $3.28 \times 4.2\,\mathrm{mm}^2$.



(a) (b)

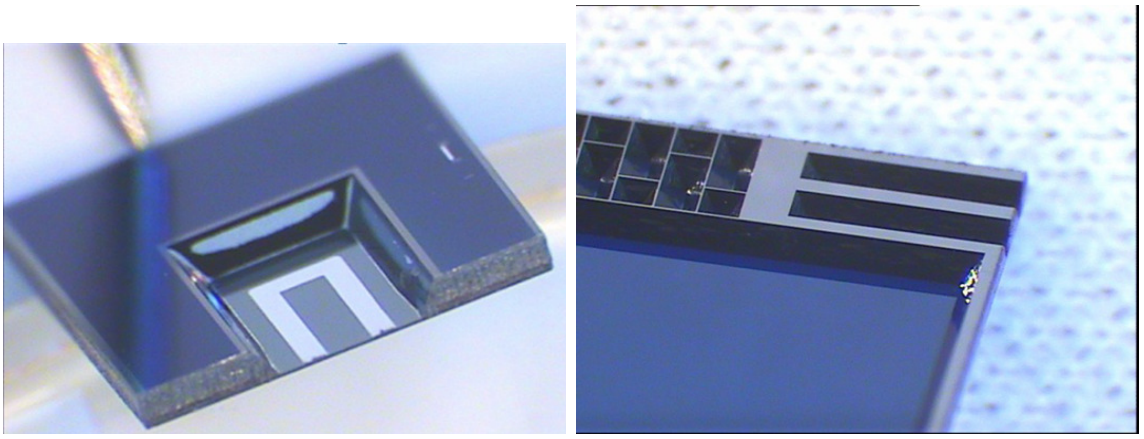
Figure 2.6.: On picture (a) the position of the 4 DHPs on a PXD module is shown. They are placed on the end-of-stave behind the DCDs. The second picture (b) shows a photomicrograph of the top side of a DHP.

2.7. PXD9 Module

A module consists of a radiation sensitive DEPFET matrix and the three different ASIC types necessary for controlling and readout. There are also all the mandatory connections between the chips and the matrix pixels. It is worth to mention, that all the conducting lines and support structures are processed on the very same silicon wafer, where the DEPFET pixels are implemented. The metal routing is done there as well as the contacts for the ASICs. For stability reasons a second wafer is bonded underneath the first one and etched down at the sensitive area, so that a thickness of just 75 µm is achieved. This ensures that the material budget in the sensitive area is very low, only about 0.2% of the radiation length [11]. For the same reason also the electronics (ASICs) are not placed on the pixels like for CMOS: Complementary Metal-Oxide-Semiconductor, a technology for constructing integrated circuits (CMOS) sensors, but on the edges.

The side of a module, where the DCDs and DHPs are placed, is called end-of-stave. There are also the connections for a kapton cable, which is soldered and wire bonded to the module. Instructions to the electronics are sent via this connection and the data from the sensor is transferred to the DHE.

There are four slightly different designs, distinguishing between the inner and outer layer and between a forward and backward direction of the accelerator system. So a module of the forward direction can be attached to its corresponding one of the backward direction resulting in a so called ladder. The 8 ladders of the inner layer have a width of 15 mm and a sensitive length of 90 mm, the 12 outer ladders have the same width, but a longer sensitive length of 123 mm. The 20 ladders, which build together the whole PXD, sum up to a total number of 7.68 million pixels.



(a) (b)

Figure 2.7.: Micrographs of a PXD6 module. (a) Section thinned down to 50 µm. Also visible is the thick (∼450 µm) supporting frame. (b) Grooves in the Silicon frame will be used for joining two modules to compose a ladder. [14]

2.8. DEPFET Handling Hub

The DEPFET Handling Hub (DHH) is the next part of the readout and control system. Due to space limitations there are no ordinary cables mounted on the PXD modules for the connection to the DHH. Flat kapton cables, which are specially designed for this purpose are used instead. A kapton cable carries the signal and power lines in several planar layers to a patch panel. The slightly different designs have a length of 43 – 47 cm. On this patch panel the lines are routed in a way to fit to the endings of ordinary cables, which transfer the data further to the DHE and connect the power lines to the power supply unit [4].

The wires of the data and power cables will be soldered onto the patch panel, again due to space limitations. During preparation and testing, patch panels with connectors are used for more comfort and flexibility. The first design used two Infiniband cables as data connection between patch panel and DHE. One Infiniband cable provides 8 shielded twisted pair copper cables for 8 LVDS signal lines. For the new design one Infiniband cable is replaced by a RJ45 cable, which is used for the 4 slow control JTAG lines. This is done in order to save costs and also space.

The DHE as shown in Figure 2.8 is a subunit of the DHH system. There are 40 DHEs - one for each module - which receive the data from the DHPs. Five DHEs are grouped together. They collect their data in one Data Handling Concentrator (DHC), which serves as a sub-event builder, and sends the processed data to a compute node via optical links [15]. The estimated total data rate for the complete PXD is ∼58 GBit/s [4].

The DHH fulfills also the task of generating the DHP system clock from the Belle II clock distribution system, so that the detectors work in sync. It also decodes the slow control

Figure 2.8.: DHE Version 3.2 with removed RAM. On the left side there are the Infiniband and RJ45 connectors to the PXD module and on the right side is the connector to the rag. [16]

signals from the external system and communicates them to the DHPs, by providing a JTAG master.

The PXD data is further processed, analyzed and combined with the data from the other detectors. A track finder and track fitter calculates which signal of the PXD is of physical interest. With this information a region of interest is built and just the data from this region is saved [4].

2.9. Production Processes and Quality Assurance

The wafers are processed at the Semiconductor Laboratory of the Max Planck Institute (HLL) in Munich. All production steps and the required semiconductor technology including wafer bonding, doping, polishing, processing the DEPFET structure, implanting the metal layers and thinning as well as cutting the modules is provided by the HLL.

In between these steps, tests on wafer level are performed several times. A needle card is used to contact the pads of test structures, which are implemented in the layout and are removed via etching after the tests. Theses tests measure the properties of the matrix and can find faults in the circuitry, e.g. opens or shorts between signal lines. At this moment the circuitry of the modules is not yet complete. The ASICs and passive Surface-Mount Devices (SMDs) like resistors and capacitors are still missing. [10]

After cutting the modules out of the wafer, they are send for further production steps to the Fraunhofer Institute for Reliability and Microintegration (IZM). The IZM takes over the part of mounting the ASICs onto the modules. As seen in figures 2.4 (b) and 2.5 (b) the pins of the ASICs are arranged in a so called Ball Grid Array (BGA). This kind of

pin arrangement is often used for designs with high limitations in space, because the whole area of the chip can be used to place the pins, instead of just the boundary as for classical designs. These BGA chips are mounted with special processes called bump bonding and flip chip mounting. At first little solder balls (bumps) are placed on the pins of an ASIC. Thereafter the chip is turned upside down (flipping) and placed on top of its footprint, the corresponding contacts on the module. Then a temperature cycle is applied to melt the solder balls. At this step the wet solder causes a self alignment and a small deviation in the placement of the ASICs is evened out. [4]

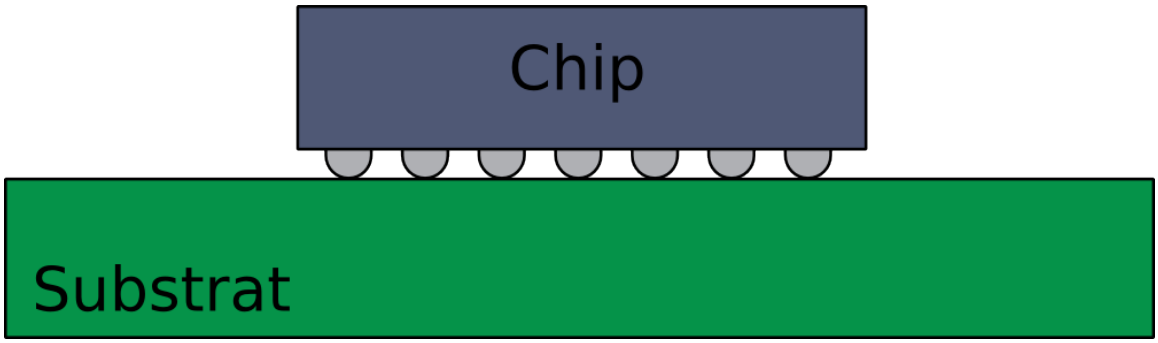


Figure 2.9.: Illustration of the connections between a BGA chip and the substrate after the flip chip process. [17]

After the bump bonding, the modules are send back to the HLL, where the passive SMD elements are mounted using a tool from finetech. The last step is the attachment of the kapton cable. It is soldered and wire bonded to the pads at the end-of-stave of the module. Bump bonding, SMD mounting and kapton attachment are done at decreasing temperatures to assure that the previous steps are not affected.

In conclusion, the assembly of a module is also a very complex process, where also faults can occur, which have influence on the performance of the system. Again tests are carried out to verify the success of the single steps. The modules undergo for example a x-ray inspection after the bump bonding of the ASICs. In figure 2.10 such a x-ray image is shown. The solder bumps are clearly visible as dark spots. Possible faults like missing bumps or a fusion of two or more bumps can be detected. This would indicate a missing electrical link (open) or shorts in several nets respectively. Unfortunately no definite statement about the electrical connection can be made, because the connection can be faulty although there is no indication in the x-ray image.

To test the actual electrical connections physical contact is required. This can be done with thin needles, which are placed on the accessible pads. But here we reach immediately the limits of this method, because all ASIC pads are buried in between the ASICs and the silicon substrate. Also most of the nets don't come up to the surface, but are in the lower layers.

Functional tests of the whole module are possible and done as well but the fault analysis in a complex system is very difficult. A fault can have many different and also multiple reasons at once. Therefore a lot of experience and knowledge is required to

interpret the results of functional tests and they are only possible after all production steps.

A better solution is the use of boundary-scan. The method and its advantages is described in the following chapter.

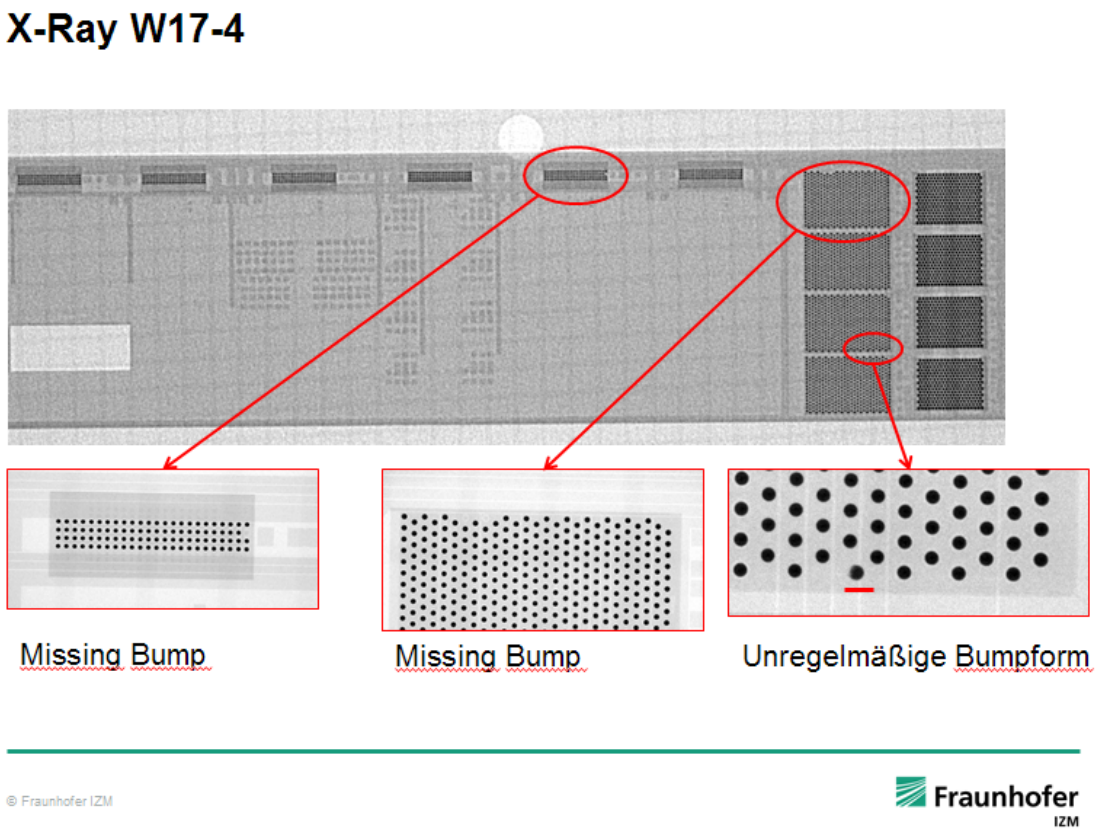


Figure 2.10.: X-ray image of the W17-4 module after bump bonding of the ASICs by IZM.

Part II.

Test System for JTAG Boundary-Scan

A fundamental law of computing: *Garbage In, Garbage Out.*
Kenneth P. Parker

3 Boundary-Scan

Boundary-scan is a software based testing tool, initially developed to solve the testing problems which arise as a consequence of more and more complex and smaller designs of digital electronics. The trend of minimization and microintegration is still proceeding and driving traditional testing methods to their limits. Therefore, the boundary-scan standard provides a powerful solution, which can keep up with this development, because it is implemented directly in the chips' logic. It must be considered and implemented by the designer of a new Integrated Circuit (IC) and thereafter the standardized design rules guarantee that its testing mechanisms can be used independently from the chips' purpose in a board design. Special boundary-scan software uses information about the actual boundary-scan implementation of a chip and the board design to calculate digital test patterns and to execute tests.

3.1. Motivation to use Boundary-Scan for PXD Modules

The use of boundary-scan provides a lot of advantages and possibilities for the quality assurance during the production processes of the PXD modules.

+ In fact, it is possible to regain electrical access to all digital ASIC pins, which are considered in the chip design to get an internal boundary-scan cell.

+ The additional logic is controlled by just 4 extra data lines, corresponding to 4 additional pins for each ASIC. Therefore, it is also easy to perform boundary-scan tests on the modules before the kapton cable is attached. In principle, a needle card needs to contact only the power pads and the 4 boundary-scan data lines.

+ Another favorable fact is that a boundary-scan test is very fast. The pure interconnection test of a whole module lasts less than 2 seconds, so that the preparation time for the measurements (approximately 10 minutes per module) dominates the testing capacity.

+ But the main advantage is of course the meaningfulness of the test result, because of its definitive statement about the electrical connections and communication between the ASICs.

+ Moreover, after the first purchase of a boundary-scan system all this is achieved with basically no additional costs. The cost differences for the production of ASICs with or without boundary-scan implemented and the costs to route 4 additional lines can be neglected. Follow-up costs after a redesign of the layout, like for a new bed-of-nails, are only present when the changes also affect the pads for the needle card.

! The only restriction is that the boundary-scan frequency is much slower than the one of the operating system. The test frequency can be varied. It is typically within the range of some MHz. For the PXD modules a frequency of 1 MHz is used. So all results are valid for the applied frequency; possible cross talk issues, which occur at higher frequencies, can not be detected.

− The only drawback of boundary-scan in general is a increased design time. The layout must consider the additional data lines and provide a contact possibility. Also the ASIC designers have to carefully follow the design rules. Otherwise an external boundary-scan tool will have troubles to reach optimal performance. For the PXD modules, however, a JTAG interface was already implemented for configuration of the internal registers of the ASICs.

Therefore, the use of boundary-scan for quality assurance during the series production of the PXD has a lot of advantages and will help to sort out defect and damaged modules before they undergo further production steps. With the detailed fault analysis of a boundary-scan test, it is also possible to decide if it is feasible to rework a module e.g. by replacement of a faulty ASIC.

3.2. Development of the IEEE Std 1149.1 (JTAG)

The earlier mentioned problems for traditional testing methods drove the leading semiconductor manufacturers of Europe and North America already in the 80s to find a new testing solution. In 1985, they got together and formed the so called Joint Test Action Group. The abbreviation JTAG became a synonym for the standardized design rules which were developed by that group and confirmed as an Institute of Electrical and Electronics Engineers (IEEE) standard in 1990. After several additions and revisions the actual version is the IEEE Std 1149.1-2013. The further developments address special possibili-

ties, which were not foreseen in the initial standard. The IEEE boundary-scan standards are:

1149.1	for digital signals
1149.4	for analog signals
1149.6	for testing advanced I/Os, like AC coupled differential signals
1149.7	for a new port definition with just two signal lines
1532	for in-system configuration

The last one (1532) got a new number, because it does not directly address the initially purpose to solve testing problems, but takes advantage of the boundary-scan access port to define a possibility of programming ICs. It is very popular to use JTAG as programming interface nowadays. Therefore, it is often better known for its possibility to configure and program ICs than for its testing abilities.

3.3. Boundary-Scan Architecture

An intrinsic part of boundary-scan is that the additional logic hast to be implemented into the logical system of an IC. The boundary-scan architecture must therefore be considered by the designer during the designing and development of a new chip. In principle, it is put around the system logic and must be invisible when the chip is operating and performing its actual task. So it has no influence on the chips' performance. The only exception are data registers, which are internal registers in the chip logic and can be used to program it and to set it in different states.

It is crucial that the design rules are strictly followed. The advantage a standard is that systems can work immediately together, although they have no other information about each other but the common standard. Kenneth P. Parker writes on the first side of his "The Boundary-Scan Handbook":

> I have twice stated that software would be utilizing the standard. In complex designs where testing problems are most difficult, Boundary-Scan is quite tedious for a human to program manually. The attendant serialization of test data makes the purpose of a test quite incomprehensible. Thus, it is extremely important that the rules of this standard be strictly obeyed, and, that the details of how a given IC has Boundary-Scan implemented be described with complete accuracy. If this warning is not heeded, then software may well obey a fundamental law of computing: *Garbage In, Garbage Out.* [18]

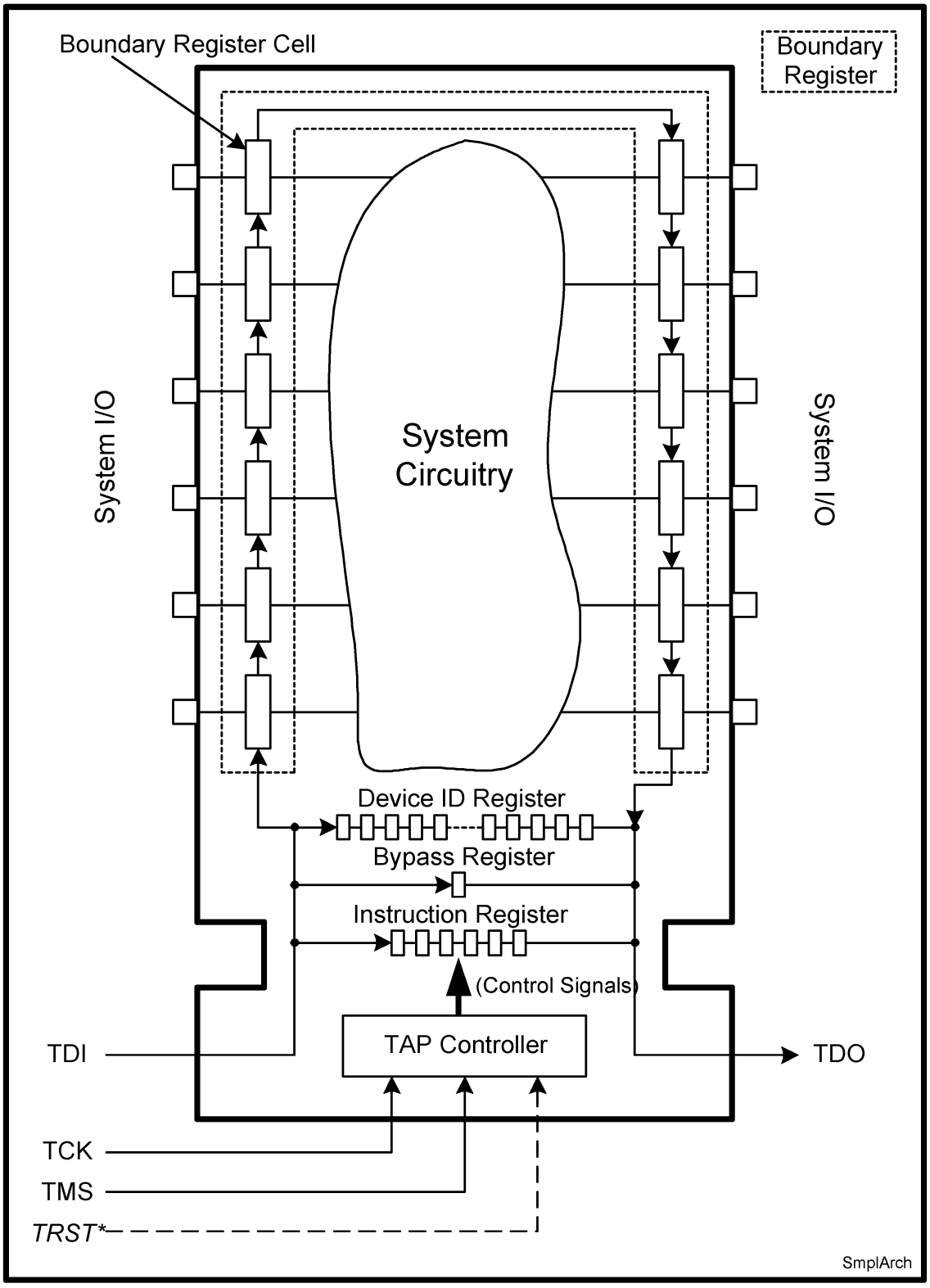


Figure 3.1.: General, simplified architecture of an 1149.1 compliant Integrated Circuit [18].

3.3.1. Test Access Port

The interface for the communication and controlling of a boundary-scan ASIC is called Test Access Port (TAP). It consists of the following 4 (optionally 5) signal lines, as shown in Figure 3.1:

TCK	Test Clock
TMS	Test Mode Select
TDI	Test Data In
TDO	Test Data Out
TRST	Test Reset (optional)

3.3.2. TAP Controller

The test protocol is driven by TCK and TMS (and TRST if present). These signals are directly connected to a finite state machine, called TAP controller. It receives the external signals and generates the control signals for the other internal boundary-scan logic. According to the standard, a transition to the next state always occurs on a rising edge of TCK. This is important, because other operations, following a state transition, are executed at the falling edge of TCK [19, 18]. TDI and TDO are connected to the starting and ending point of the boundary-scan relevant shift registers. The data is serially shifted in and out again; and the TAP controller decides what happens with the data in between.

To reset the TAP controller either TRST can be used or a defined reset sequence. After 5 cycles of TCK, during which the TMS is held high, the state machine will always be in the reset state, independently of its initial position [18].

The state transition diagram in figure 3.2 illustrates the possible paths through the finite state machine, which all JTAG applications have to follow. The arrows between the single states indicate, which logical value must be set for TMS before the rising edge of TCK to reach the next state or maybe stay in the current one. It is clearly visible, how almost all of the states group into two vertical columns. The left one is referred to as data column, the right one as instruction column. Both are exactly identical apart from one letter in their labels. So if one enters one of them, their behavior and therefore the actions performed by the TAP controller to the corresponding register are the same. For a better understanding of this sixteen-state machine, a simplified example of a path through it is given below.

But first the different registers, which are controlled by the TAP controller, are described. As indicated by the two columns, there are two different kinds of registers: the Instruction Register (IR) and the Data Registers (DRs).

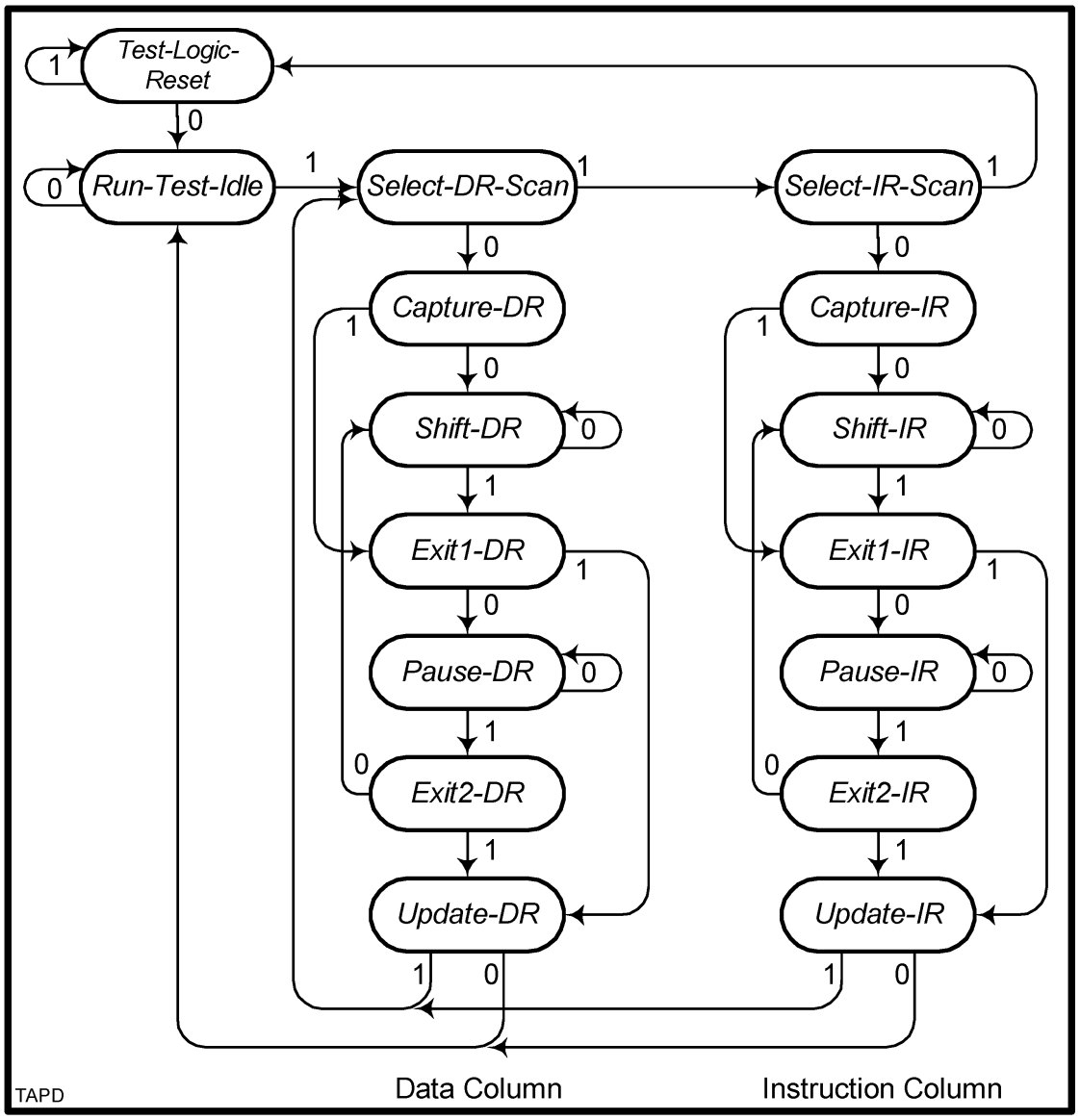


Figure 3.2.: State transition diagram of the sixteen-state TAP controller [18].

3.3.3. Instruction Register

The IR consists of a shift register with a minimal length of two bits and of a parallel hold rank of the same length. The actual length determines the number of possible instructions. All instruction codes must have the correct length. The shift register cells can either be updated one by one, through shifting in data from the TDI during the *Shift-IR* state of the TAP controller, or all at once from a parallel capture register, which contains a specific instruction-capture code. The values of the shift register are loaded into the parallel hold rank during the *Update-IR* state. The instruction code in the hold rank determines the operational mode of the boundary-scan logic.

3.3.4. Data Registers

Two DRs are mandatory by the IEEE Std 1149.1: the bypass register and the boundary-scan register. There are several optional DRs as well. Some of them are also suggested and described in the standard. The active instruction code determines which DR is put in between the TDI and TDO and how the data is processed.

Bypass Register

The mandatory bypass register consists only of one cell. As indicated by the name, when the bypass register is chosen, the way from TDI to TDO is shorted, bypassing all other shift registers. This is done to reduce testing time, when other chips are under test. Furthermore, as standard value for the single register cell a “0” is loaded, when the TAP controller switches to the bypass register. This is connected to the obligatory “1” for the Least Significant Bit (LSB) of the ID-Code.

Device Identification Register

The Device Identification Register is not mandatory, but when implemented, the ID-Code and therefore the ID shift register must have a length of 32 bits. There is a scheme suggested by the standard for the assignment of an ID-Code, including version, part and manufacturer’s number [19]. Besides the defined length, the value of the LSB is obligatory, too. It must be set to “1”. In combination with the default “0” of the bypass register, this allows for a *blind interrogation* of a chain of components. When the TAP controller is in the *Test-Logic-Reset* state, the ID-Code instruction is automatically loaded into the IR if the component has the corresponding register. If not, the bypass instruction is loaded instead. When a JTAG software leads the TAP controller from the *Test-Logic-Reset* state directly to the data column and shifts out the whole chain, it is possible to count the devices in the chain and to assign them their ID-Codes. A “0” indicates that this ASIC has no ID-Code implemented and the next bit will be considered as coming for the next ASIC. A “1”,

however, announces an ID-Code and the next 31 bits will be interpreted as the remaining part.

Simplified example for an ID-Code readout

The meaning of the state diagram and the processes in the TAP controller can be understood more easily by following a simplified example for a boundary-scan task. The ID-Code of an ASIC is read out to check if the correct one was mounted at the correct position. After powering up the ASIC, the TAP controller should be in the *Test-Logic-Reset* state. This assures that the system of the IC is not affected by the boundary-scan logic and can operate properly. However, a JTAG software will always apply a reset at the beginning of communication. This can be done asynchronously by the TRST. But although all ASICs, which were developed for the PXD modules, are sensitive to this signal, it can not be used, because it is not connected to the outer control electronic. The second reset possibility is used instead: TMS is hold high for 5 TCK cycles.

At first it is necessary to enter the instruction column. This is reached by a TMS sequence of:

0 The *Test-Logic-Reset* state is left into *Run-Test-Idle*. There is the possibility for self-tests of the IC's circuitry, if implemented and as long as this state is occupied. But now it is considered as idle. With a

1 the *Select-DR-Scan* state it entered and now it is possible to either enter the data column by a 0 or to leave it with a

1 to the *Select-IR-Scan* state. Now the instruction column is chosen with a

0 When the *Capture-IR* state is entered, a value, predefined by the chip design, is parallel loaded into the IR. The last two significant bits must be "01". The remaining bits (number depending on the IR length) can adopt an arbitrary pattern.

The instruction-capture value of the Switcher for example is "101". This bit pattern should be communicated by the designer. Than it is possible to verify the integrity of the JTAG chain with it. The instruction-capture code can be shifted out and compared to the expected value.

To reach the ID-Code, some further steps are missing. It is necessary to load the correct instruction into the IR. This is done after entering the *Shift-IR* state. As long as the TAP controller is in this state, the first cell of the IR is connected to the TDI and the last one to the TDO respectively. As long as the TMS is held low, bit after bit is shifted into the IR. Hereby, the LSB is the first bit entering the shift register. The JTAG software needs to know the correct instruction code to choose the ID-Register mode. In the example, the correct instruction code for the Switcher is "010".

After the instruction code is shifted into the IR, the TMS is pulled up again, so that the TAP controller leaves the *Shift-IR* state. TDI and TDO are disconnected from the IR. The rest of the column is skipped and the *Update-IR* state reached by another "1" on TMS. In this state, the values previously shifted into the IR are updated to the parallel hold elements of the IR. A new operational mode is set.

To get to the ID-Code, the data column is selected with a "10". Due to the stored value in the IR, all following actions concern the ID-Code register. In the *Capture-DR* state the 32 bit long ID-Code is loaded into the corresponding ID-Code shift register. By entering the *Shift-DR* state, the TDI and TDO are connected to this ID-Code shift register. As long as the TAP controller stays in this state, the bit pattern in the ID-Code shift register is shifted out to the TDO. The values shifted into the ID-Code shift register remain meaningless because they can not update the ID-Code value and are overwritten the next time when a *Capture-DR* is applied on the ID-Code register. But the values shifted out are of particular interest. The JTAG software is looking for a special ID-Code and can verify in this way that the correct ASIC is mounted in the correct place in the circuitry. At least a reset is applied to bring the ASIC back to an operational state.

Summing up the logical values for TMS, TDI and TDO for the whole measurement, we get the following table. The x indicates that the value is not relevant for the interpretation of the readout. The JTAG software is generating all test signals and comparing the results to the expected values. It is also able to analyze occurring faults and to suggest possible reasons.

	reset TAP ↓	enter IR ↓		shift IR ↓	enter DR ↓		shift DR ↓	reset TAP ↓
TMS:	11111	011	00	000	111	00	00000000000000000000000000000000	11111
TDI:	xxxxx	xxx	xx	010	xxx	xx	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx	xxxxx
TDO:	xxxxx	xxx	xx	101	xxx	xx	00100011010001010110011110001001 ↑ ID-code	xxxxx

User-Defined Registers

The ASIC designers have also the option to use the infrastructure provided by the JTAG standard to implement own, user-defined data registers. The only restriction is that they must form a consistent path between TDI and TDO, to not brake the chain. Than they can be used to configure the system logic, for example. The corresponding instruction codes and the information, how to use the user-defined registers, must be communicated in full detail by the designers.

3.3.5. Boundary Register

The most important register is the boundary register. With this register it is possible to check the surroundings of the ASIC and to communicate with the board circuitry. The boundary register consists of one or more Boundary-Scan Register Cells (BSCs), which are placed immediately behind the system's Input/Output (I/O) pins. In fact, this register is used to control the logical states of the output pins and to sense the values at the input pins. The maximal number of BSCs, which can be implemented in an ASIC, is, according to the 1149.1 standard, equal to the number of digital I/Os of this ASIC minus four (or five) for the TAP signals, which must be excluded. The length of the boundary-scan shift register can be longer because there can be more than one shift register cell in a BSC, depending on its design.

Boundary Register Cell

The basic design of a BSC is shown in Figure 3.3. "Shift In" and "Shift Out" are the connections to the previous and following BSC. In this way, the test data can be shifted in from the TDI through the whole register chain and out to the TDO. "Parallel In" and "Parallel Out" are the links to the surrounding world, where measurement data can be sensed or output pin values can be written. The simplified schematic shows a BSC built of two multiplexers and two flip-flops, which can be used as input as well as output cell. For an input cell the "Parallel Input" would be connected to the ASIC pin and the "Parallel Output" to the system circuitry, for an output cell vice versa.

However, independent from where the signal enters the BSC, it is passed through without any interaction, when the multiplexer on the right side is open for the "Parallel In". Also with no influence on the system data the left multiplexer can pass the signal from the "Parallel In" to the Capture Flip-Flop. This can be done simultaneously at all BSC. A "snapshot" of the actual values at the I/Os is taken and stored in the Capture Flip-Flops. If the left multiplexer switches to the "Shift In", all Capture Flip-Flops are connected to form the boundary shift register and the snapshot can be shifted out.

In the same way, artificial values can be shifted in from the TDI and transferred to the Update Flip-Flops, which together form the parallel hold rank of the boundary register. After the right signal is applied to "Mode", the right multiplexer switches to the signal coming form the Update Flip-Flop. Now the user/software defined values are written to the output. These are the mechanisms to regain electrical access to the physical blocked ASIC pins.

The real designs of BSCs are much more complex. For example, it is possible to create bidirectional BSCs by combining three of the mentioned basic cells: one serves as input cell, one as output cell and the third as control cell, which enables the output divers. There are a lot of more complex designs for special applications thinkable [18], but the ASICs for the PXD modules use only "just input", "just output" and control cells.

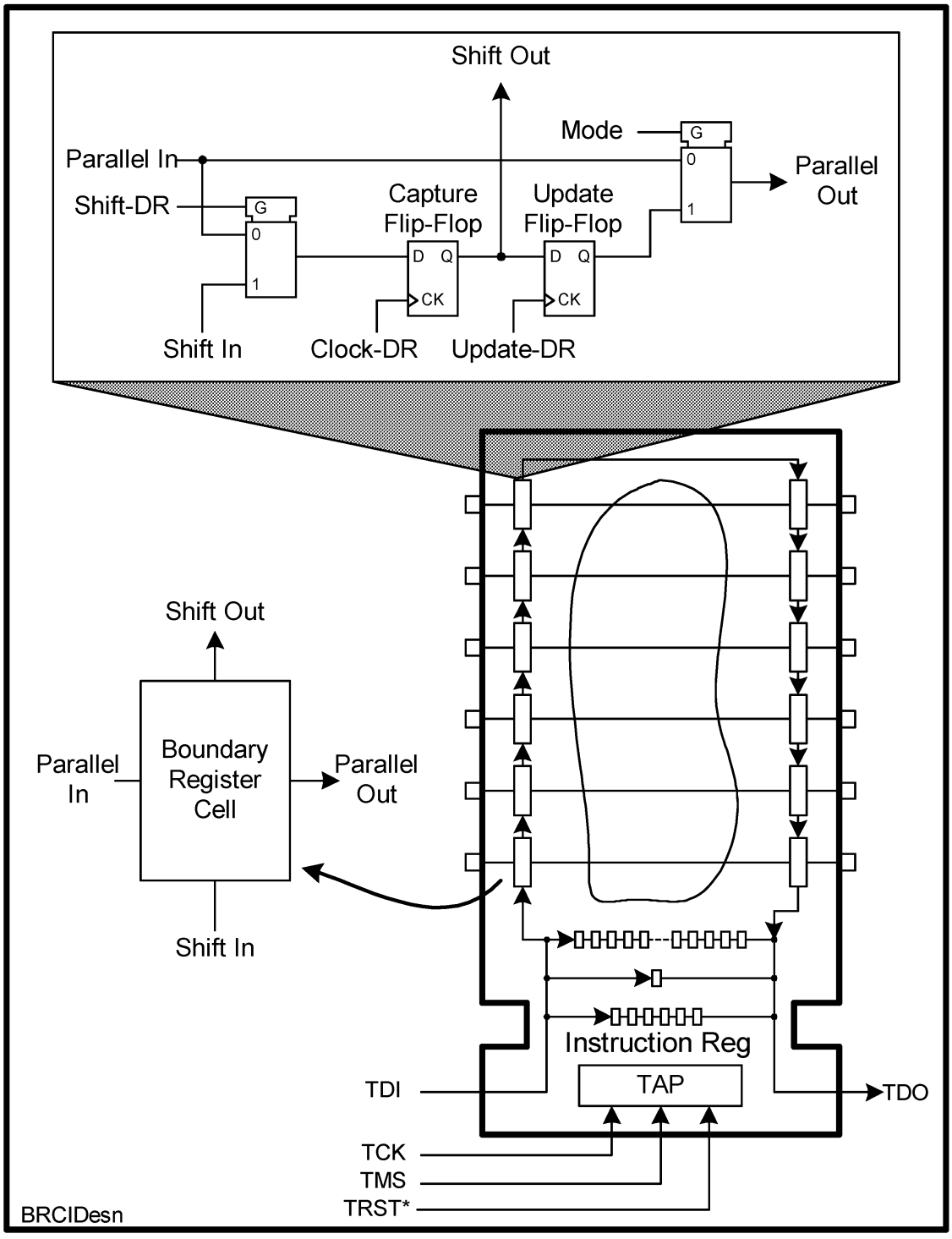


Figure 3.3.: A Typical Boundary Register Cell [18].

3.4. Operational Modes

The instruction codes define not only which DR is put in between TDI and TDO, but also set the operational modes. The operational mode determines which actions are performed with the selected DR.

The **BYPASS** and **IDCODE** modes are explained in the example above. They do not affect the system circuitry, therefore they are called non-invasive. Both modes refer to the corresponding DR. There is just one operational mode for each of these two registers.

The boundary register can be operated in several modes. The mandatory **SAMPLE** mode puts the boundary register in between TDI and TDO. The only action allowed is to sense the BSC's input and store it in the Capture Flip-Flop. Therefore, it is an non-invasive mode. The taken snapshot can be shifted out.

The **PRELOAD** mode is also mandatory. It refers to the boundary register as well. Test data can be shifted in the boundary shift register and updated to the hold rank, when the TAP controller passes the *Update-DR* state. But the output multiplexer is not switched to the Update Flip-Flops, so this is also a non-invasive mode. It is used to load the data properly before the output is enabled.

There are also several invasive operational modes, but just one of them is mandatory: the **EXTEST**. Invasive means, that the system of the chip is affected in terms of a disconnection of the I/Os pins from the system logic. This is necessary to ensure that an output pin is not driven against a system value. The disconnection, however, can be a serious problem for the internal system. A disconnected reset line, for example, could lead to a undefined state after a radical cut at the systems interface. The designer has to keep this in mind and could compensate this e.g. by adding some extra logic to hold specific values.

The **EXTEST** mode puts again the boundary register in between the TDI and TDO. The values stored in the Update Flip-Flops during the previous **PRELOAD** are now written to the outputs. At the same time, the input values are captured in the Capture Flip-Flops. When the TAP controller is in the *Shift-DR* state, the captured values are shifted out and new output values for the next test are shifted in. "EXTEST is the workhorse of Boundary-Scan testing." [18]

There are several further invasive modes like INTEST, RUNBIST, HIGHZ and CLAMP. They are all optional and suggested for special testing situations [18]. Non of them is implemented in the DCD or Switcher and the DHP has only the INTEST. Therefore, they are not important for the boundary-scan tests of the PXD modules.

3.5. How does a Boundary-Scan Test work?

The idea behind all these standardized implementations is to have two or more boundary-scan ASICs on one board[1]. Then it is possible to test the connections between them. Usually all ASICs are connected to form a simple chain (see Figure 3.4). They all share TCK and TMS signals, which means that all TAP controller are always in the same state. The shift register leads from chip to chip by connecting the TDO pin of the first to the TDI pin of the second one and so on. It is also possible to split long chains in two or more smaller chains with individual TAPs or to create chains which share just a part of their TAP. This can be done to save shifting time, but increases the complexity and sets higher demands on the software.

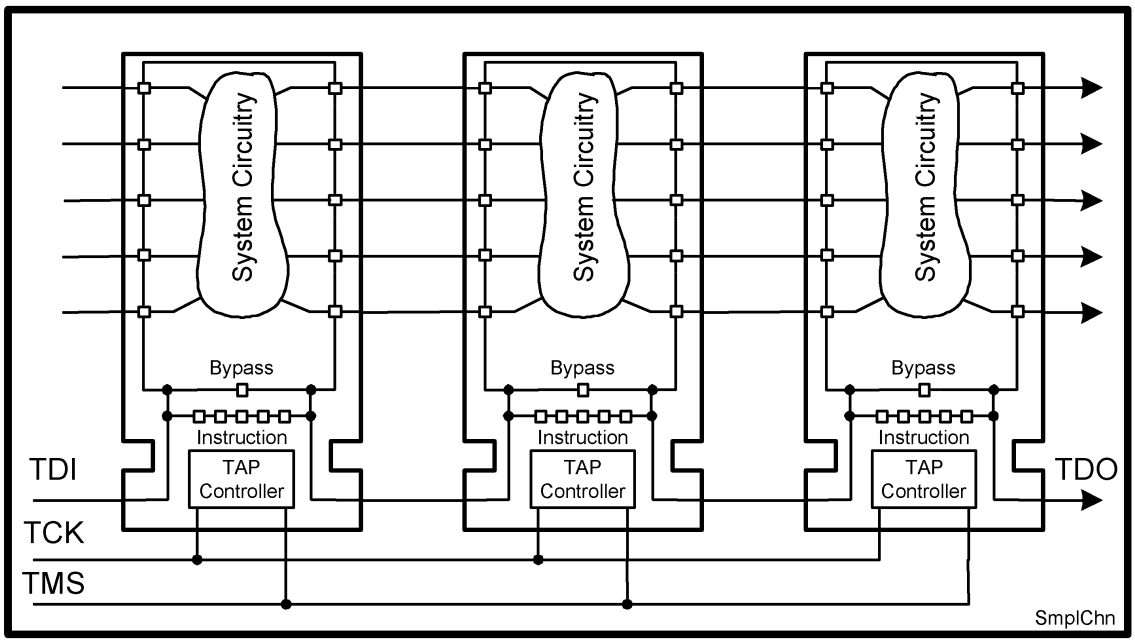


Figure 3.4.: A simple chain of Boundary-Scan ICs [18].

The first test is always to check the integrity of the chain. This is done by loading the instruction-capture codes into the IRs and shifting them out. The measured values are compared to the expected ones. The standard requires "01" as the last two bits of an instruction-capture code. Therefore the received data must toggle at least at that points. If a short, an open or a stuck is present somewhere between two ASICs, the readout will (most likely) differ from the expectation. To further improve the testing, so called sentinel bits are set in front of the values shifted into TDI. They are not intended to stay in the internal ASIC registers, but shifted right through them. The software generates them and expects to read them back at the TDO after the values of all internal registers. The second part of an integrity check is the verification of the ID-Codes. If present, the ID registers are loaded and shifted out. It is possible to check if a pin-compatible ASIC, maybe an older version, was mounted instead of the new one.

[1] It is an actual discussion how much can be achieved by just one boundary-scan capable device on a board [20].

When the integrity of the chain is confirmed, it is possible to start with further tests. The most important one is the interconnection test. Usually, the components on a board are meant to work together. Therefore, they must communicate and are linked by interconnections. Exactly these lines are investigated by the interconnection test. If an output pin of one chip is connected to an input pin of another and BSCs are implemented at both, a JTAG software can gain access to them and try to read back the value at the input pin which it wrote at the output pin. This is exactly what the ASICs will do during operation. But the JTAG software knows both ends of the lines and can easily state if a communication problem refers to a fault in the interconnections.

3.6. Reliability

Although JTAG provides a solution to find faults in electrical circuitry, one might wonder how sensitive is it for own faults. However, there is no literature about the reliability of JTAG. The experience during this work showed a high reproducibility and stability of the used systems. All faults, detected by boundary-scan tests, were verified by functional tests. The modules which passed the tests showed no problems, which could point to a faulty circuitry. The problems which occurred during testing were mainly related to operator errors. Mostly a cable was not or not properly connected.

Beside cabling, a lack of information about the boundary-scan implementation and deviations from the standard were obstacles for the JTAG system. A lot of suggestions for Design for Test (DFT) are made in [18], which especially aim to improve test coverage. An increase in test coverage opens more connections to the access of boundary-scan. Because the untested region decreases, the statement of a boundary-scan software about the functionality of a board becomes more reliable.

As a software is generating and performing boundary-scan tests, the strength and possibilities of the used algorithms is defining the performance in terms of speed and reliability. But the specialized JTAG companies do not reveal their proprietary software. Also the quality of the used JTAG hardware and especially the connections have influence on the test performance, similar to the cabling.

Another point is the relative low frequency of JTAG. The DCD clock, for example, runs at full speed with a frequency of 305 MHz. The boundary-scan operations are performed with a TCK frequency of 1 MHz. Therefore, the results just state if the physical connections are fine. JTAG is not sensitive to possible cross talk between neighboring lines. But cross talk can be a problem for the system operating at full speed. The tester must be aware of the circumstances during testing and keep them in mind for the interpretation of the results.

4

BSDL Files

The IEEE Std 1149.1 states very specifically, how boundary-scan must be implemented in an ASIC design. But the exact details vary from chip to chip, as do their purposes. The IR must have a length of at least two bits, but four different instruction codes are often not enough. The length of the boundary register is also correlated to the chip design. A chip which is just digital can have at almost every pin a 1149.1 BSC; a chip with an analog part not on these pins.

A JTAG software is needed to take control over the boundary-scan implementations. It creates the TAP signals and calculates which bit pattern has to be shifted in and which is expected to come out. However, to do so, it is necessary that the specific boundary-scan implementation is known in full detail. Therefore, the ASIC designer has to provide this information in form of a file that can be read by the software. The standard also describes how this has to be done. In Annex B of the IEEE document a machine-readable language for this purpose is described, the so called Boundary-Scan Description Language (BSDL). "BSDL documentation is a mandatory requirement of this standard; components cannot claim conformance without valid BSDL documentation." [21]

4.1. Structure of BSDL Files

BSDL is based on the syntax and grammar of VHSIC Hardware Description Language (VHDL). But it has its own structure and several constructs, which deviate from VHDL, because its purpose is a different one and focused on the description of boundary-scan implementations [21].

One example for a VHDL statement, that is used in BSDL, is the entity statement. Each description of a component is enclosed by

```
entity <component name> is

       {BSDL statements to describe the entity}

end <component name>;
```

The expression `<component name>` is a VHDL identifier and has to be replaced by the name of the individual chip without the brackets. In fact, identifiers must start with a letter [21]. Everything in between is called the entity body. It contains several mandatory and some optional statements, which have to occur in a specific order. An exemplary list is given in [18]:

```
entity <component name> is
   <generic parameter>
    <logical port description>
    <standard use statement>
   {<use statements>}
    <component conformance statement>
    <device package pin mappings>
   {<grouped port identification>}
    <scan port identification>
   {<compliance enable description>}
    <instruction register description>
   {<optional register description>}
   {<register access description>}
    <boundary-scan register description>
   {<RUNBIST description>}
   {<INTEST description>}
   {<BSDL extensions>}
   {<design warning>}
end <component name>;
```

The statements in "{ }" are optional. The mandatory statements and the optional `<grouped port identification>` and `<optional register description>`, which are used for the ASICs of the PXD modules, are explained below.

4.1.1. Generic Parameter

It is not unusual that a specific IC logic is placed in different physical packages, with different pin configurations. In this case, it is not necessary to create more BSDL files. The different pin configurations can all go in the same file but it must be possible to choose the correct one.

The generic parameter is a string with the name PHYSICAL_PIN_MAP. It can be set to a default value:

```
generic (PHYSICAL_PIN_MAP: string := "BGA_15x15");
```

In this example, the package called BGA_15x15 is set as default. If the JTAG software wants to use the configuration of another package, it can fill the generic parameter with the corresponding value.

4.1.2. Logical Port Description

In the logical port description, logical names are assigned to the I/O pins. They must not be confused with the physical pin identifiers, which are often numeric or alphanumeric (for BGA packages). All I/O pins of an ASIC should be described here, including the TAP pins as well as all pins not correlated to boundary-scan.

```
port (<port name>    : <pin type>    <port dimension>;
      TDO            : out           bit;
      TMS, TDI, TCK  : in            bit;
      Digital_Out    : out           bit_vector(1 to 8);
      Digital_In     : in            bit_vector(1 to 8);
      GND, VCC       : linkage       bit);
```

Here the `<port name>` can be chosen to be as specific as possible. The `<port dimension>` is either `bit` for a single pin, or `bit_vector` for a whole bus. The possible values for `<pin type>` are shown and explained below:

`<pin type>`	explanation
`in`	input-only pin
`out`	output-only pin with 3-state driver (high, low, highZ)
`inout`	output-only pin with 2-state driver (high, low)
`buffer`	bidirectional pin
`linkage`	others like power, ground, analog, no-connect

4.1.3. Standard Use Statement

The standard use statement refers to a VHDL package that contains definitions of BSDL statements. It instructs a VHDL analyzer, while reading the file, to search there for the definitions of the statements which occur in the following description. Also user defined use statements, which refer to an user defined package, are possible. Note in the example that a comment in BSDL is placed behind -- characters.

```
use STD_1149_1_2001.all; -- Get Std 1149.1-2001 definitions
```

4.1.4. Component Conformance Statement

The component conformance statement identifies which release of the standard was used to design the boundary-scan circuitry within an IC.

```
attribute COMPONENT_CONFORMANCE of <component name>: entity is
    "STD_1149_1_2001";
```

4.1.5. Device Package Pin Mapping

In this section, the logical port names are mapped to the physical pins. It is possible to describe multiple mappings because the attribute PIN_MAP can have more than one constant PIN_MAP_STRING. The JTAG software can call the required mapping with the correct constant.

Typically the number of pins is too large, so that the string with all assignments does not fit in one line. The parts of a string can be concatenated with the one in the next line with the & operator.

```
attribute PIN_MAP of <component name>: entity is PHYSICAL_PIN_MAP;

constant BGA_15x15: PIN_MAP_STRING:=
     "CLK: A1, Digital_Out: (B2,B3,B4,B5,B7,B8,B9,B10), " &
     "Digital_In: (C23,C22,C21,C20,C19,C17,C16,C15), " &
     "GND:A6, VCC:A18, " &
     "TDO:F11, TMS:F12, TCK:F13, TDI:F14";

constant leads_225: PIN_MAP_STRING:=
     "CLK:9, Digital_Out: (10,11,12,13,16,17,18,19), " &
     "Digital_In: (6,5,4,3,2,27,26,25), " &
     "GND:14, VCC:28, " &
     "TDO:20, TMS:21, TCK:23, TDI:24";
```

As shown in the example, the pin mapping of data buses is done with grouped pins in brackets. The `Digital_Out(1)` of the leads_225 package is mapped to pin 10. For chips with matrix like pin arrangements (e.g. BGAs), the pins are usually numbered alphanumerically. Unlike the examples, where for simplicity reasons just a fraction of all system pins is shown, it is highly recommended that every pin of an ASIC is considered and described in the BSDL file.

4.1.6. Grouped Port Identification

The grouped port identification is optional. The most common case, where it is used, is the LVDS communication. Two lines are used to carry one bit of information, therefore, they belong to each other. When they are described in this section, a boundary-scan software can control them properly and use them also for tests.

```
attribute PORT_GROUPING of <component name>: entity is
      "Differential_Voltage ((V_P(1), V_N(1)),      " &
                            "(V_P(2), V_N(2)),      " &
                            "(V_P(3), V_N(3)))";
```

It is also possible to use directional current flow for LVDS. BSDL accounts also for this with the attribute `Differential_Current`. A convention in BSDL is that for differential pin pairs always the positive "P" pin is mentioned before the negative "N" pin. For later references, a pair of differential pins is always associated with just the positive pin.

4.1.7. TAP Port Identification

The TAP port identification is needed to assign the TAP signals to the correct logical port names defined at the beginning. Here, they correspond to the common names, but they could also have different names in specific designs. The first four are mandatory. The last one (TRST) may only occur when it is implemented. For the TCK two parameters are given in the brackets. The first one is the maximal TCK frequency. The second one, which could either be `BOTH` or `LOW`, states if the TCK can be stopped just at low or both at low and at high.

```
attribute TAP_SCAN_CLOCK of TCK   : signal is (1.000000e+06, BOTH);
attribute TAP_SCAN_IN    of TDI   : signal is true;
attribute TAP_SCAN_MODE  of TMS   : signal is true;
attribute TAP_SCAN_OUT   of TDO   : signal is true;

attribute TAP_SCAN_RESET of TRST  : signal is true;
```

4.1.8. Instruction Register Description

In the instruction register description all important information about the IR and the controlling of the TAP is given. At first, the length of the IR is stated. Then the possible instructions and the corresponding instruction codes ("opcodes") are listed. The length of all opcodes must match the length of the IR. The standard requires an all "1"s code for the **BYPASS** instruction. The other opcodes do not have a preset bit pattern and must be stated. As **SAMPLE** and **PRELOAD** can be performed in one TAP controller cycle, it is possible to assign them the same opcode as in the example below. It is also possible to define more than one opcode for one instruction.

```
attribute INSTRUCTION_LENGTH of <component name>: entity is 5;

attribute INSTRUCTION_OPCODE of <component name>: entity is
     "BYPASS                    (11111)," &
     "EXTEST                    (00000)," &
     "SAMPLE                    (00010)," &
     "PRELOAD                   (00010)," &
     "INTEST                    (10001)," &
     "IDCODE                    (00001)," &
     "SOMETEST                  (10111)";

attribute INSTRUCTION_CAPTURE of <component name>: entity is "00001";
```

At the end of the section, the instruction capture code is given. The standard requires "01" to be the last two bits.

4.1.9. Optional Register Description

In the optional register description the ID-Codes of all devices are given, which are described by this BSDL file. A new version of an ASIC does not necessarily need a new BSDL file, when the boundary-scan implementation stays the same. Also, different pin packages or identical chips from different vendors (second sourcing) most probably have different ID-Codes, but can be described with the same BSDL file. The software will check if the ID-Code of an ASIC matches one of the ID-Codes given in the BSDL file before applying other tests to the chip.

```
attribute IDCODE_REGISTER of <component name>: entity is
     "01X0" &                   -- 4-bit version number
     "0100010010000101" &       -- 16-bit part number
     "00110001000" &            -- 11-bit identity of the manufacturer
     "1";                       -- Required by IEEE Std 1149.1
```

```
attribute USERCODE_REGISTER of <component name>: entity is
     "0000" & "1111" & "1111" & "0000" &
     "0000" & "0010" & "1010" & "1100";
```

More than one ID-Code can be given by simply placing a second one after the first. The semicolon must be placed after a multiple of 32 bits. Another option is the usage of the character "X", which indicates that a "0" as well as a "1" is valid for this bit. When this method is not sufficient to describe all possible ID-Codes, it is also possible to define bit patterns. All combinations of these patterns (with LSB=1) are valid ID-Codes.

4.1.10. Register Access Description

The register access description is necessary to define on which registers user-defined instructions act. It is redundant to state that **EXTEST** acts on the BOUNDARY register. In fact, all but the last information in the example below is redundant. The user-defined instruction **SOMETEST** acts on the TEST_REG register, which has a length of 144 bits. User-defined instructions can also act on standard registers like the BOUNDARY register.

```
attribute REGISTER_ACCESS of <component name>: entity is
        "BYPASS               (BYPASS)," &
        "BOUNDARY             (EXTEST, SAMPLE, PRELOAD, INTEST)," &
        "DEVICE_ID            (IDCODE, USERCODE)," &
        "TEST_REG[144]        (SOMETEST)";
```

4.1.11. Boundary-Scan Register Description

The boundary register has to be described very detailed. In fact, at least four and up to seven properties of every register cell are needed. But the first information in this section is the length of the boundary register. Thereafter, a list of all cells in decreasing or increasing order follows. For each cell the required information is given in brackets. At first, the design type of the *cell* is stated. It must be defined in the package called in the use statement, e.g. in STD_1149_1_2001.all. The second information is the *port*, which is connected to this cell. Control cells, which are not connected to a port, get an asterisk. Then the *function* of the cell is stated. The function of an input cell is of course input. A output cell can be either two-state (output2) or three-state (output3). A three-state output cell can be deactivated (high impedance) by a control cell. The *safe* value indicates which value should be loaded, when the boundary-scan software would otherwise choose an arbitrary one, so that a safe state is achieved. The last three values of an output cell show the information concerning the corresponding control cell. The number in the *ccell* column is the number of the control cell. The *disval* gives the value which must be loaded

into the control cell to disable the output cell. In the *rslt* column is the result of disabling shown.

```
attribute BOUNDARY_LENGTH of <component name>: entity is 6;

attribute BOUNDARY_REGISTER of <component name>: entity is
--
--  num   cell   port          function   safe [ccell disval rslt]
--
    "5   (BC_1,  Dig_In(2),    input,     X),                    " &
    "4   (BC_1,  Dig_In(1),    input,     X),                    " &
    "3   (BC_1,  Dig_Out(2),   output3,   X,     1,     0,    Z), " &
    "2   (BC_1,  Dig_Out(1),   output3,   X,     1,     0,    Z), " &
    "1   (BC_1,  *,            control    0),                    " &
    "0   (BC_1,  CLK,          input,     X)                     ";
```

The boundary-scan register description is the last mandatory part of a BSDL file. With these descriptions, the boundary-scan implementations of an ASIC are documented sufficiently. A JTAG software needs to load the BSDL files of all components to know how it can work with them.

4.2. BSDL File for DHP

A first version of the BSDL file of the DHP was automatically created by the ASIC design software. Two different versions of the chip were tested during this work: DHPT0.2 and DHPT1.0. Both must have their own BSDL file because there were also changes in the boundary-scan implementation. A comparative overview of all used ASICs is given in the table below.

Some changes in the BSDL files for the DHPs were necessary. During the first phase of testing the interaction of the software with the ASICs, the identification of the bussed digital pins was problematic. This was causally related to the netlist format and missing physical pin mapping, as explained below. Therefore, all pins were individually defined and buses omitted, except for linkage pins. For the LVDS signals the grouped port definition was added.

In the manual of the DHP the pins just have names as identifiers and not a number, which would state their position on the chip in a defined way [22]. Therefore, a alphanumerical numbering scheme was introduced as it is commonly used for industrial chips. As the ASIC pins are arranged in a matrix like way, continuous characters are assigned to the columns and continuous numbers to the rows (see Figure 4.1). For the DHP (and DCD), the even rows are shifted by a half of the pin distance to the left compared to the odd rows. So the matrix pattern is not so clear, but the column counting starts always at the first pin of a row with “A”.

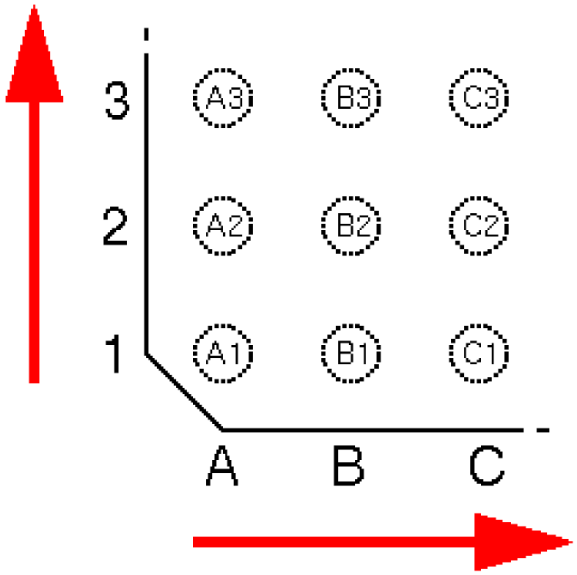


Figure 4.1.: Alphanumerical numbering scheme for BGAs. [23]

4.3. BSDL File for Switcher

The BSDL file for the Switcher was written by the responsible designer, Christian Kreidl. He used the original DHP file as template. For the current version, the analog linkage pins were added and the LVDS signals defined in the grouped port definition. The alphanumerical numbering scheme of the BGA was already given in the manual [24], but had to be transferred to the BSDL file.

4.4. BSDL File for DCD

The BSDL file for the DCD was written from the scratch, using the original DHP file as template as well and following the descriptions and suggestions of the IEEE document. The design files (Verilog files) of the DCD were analyzed to get the information about the implementation of the boundary-scan circuitry. Also a lot of try-and-error testing was necessary. As for the DHP, bussed pin definitions are only used for linkage pins. The DCD has no LVDS pins connected to BSCs, therefore, a grouped port definition is not necessary. The ID-Code section had to be commented out because the chosen ID-Code has a LSB=0 and is therefore not compliant to the standard. Also, the TAP controller changes its states at the falling edges of TCK, which is another conflict to the IEEE Std 1149.1 [25]. This is currently handled with an inversion of the TCK signal in the DHP, which provides the JTAG signals for the other ASICs. Both points were discussed with the DCD designer and are considered in the newest DCD submission.

For the pin mapping, the alphanumerical scheme was used as well. Because of the higher number of columns, there are some omissions in the characters for better legibility: I O Q S U X Z are not used, because they could be confused with other characters.

Summary and comparison of created BSDL files

As written in the IEEE Std 1149.1, without a valid BSDL file a component cannot claim conformance with the standard. The BSDL files of two of the ASICs were missing and the existing one was not complete. It was the first part of this thesis to develop valid BSDL files and prepare the ASICs in that way for boundary-scan tests. With the correct BSDL files (and some changes in the DCD), the ASICs used for the PXD modules are now conform to the IEEE Std 1149.1.

	DHPT0.2	DHPT1.0	SwitcherB18v2	DCDB4 pipeline
IR length	8	8	3	4
# of opcodes	14	18	5	6
IR capture	00000001	00000001	101	0101
ID hex	44485011	44485011	23456789	12345678
boundary register length	109	115	5	83

Consistency is better than truth.

Manfred Valentan

5 Netlist

After the development of an electrical board with several components and connectors, the netlist contains the information about the connections on this board. Usually, the lines do not connect just two pins but can have several branches as well. Therefore, a connection line with all its branches is called net. However, the physical routing of these nets is not stated in the netlist. The netlist contains the names of all nets and groups all pins which are connected to one net. All components which are integrated in the circuitry are listed with their pins as well.

The components of an electronic device are usually mounted on a Printed Circuit Board (PCB). The individual components are placed on so called footprints, which match the configuration of the pins. The lines are routed in one or more layers so that the correct pins are connected. This is done with a PC software and the completed layout file is sent to a PCB manufacturer, who will produce the required number of boards. The finished PCBs must then be populated with the electrical components by soldering them onto their footprints. The procedure for the all-silicon PXD modules, however, is completely different (see section 2.7). The DEPFET matrices are not cut out of the wafer to be placed on a PCB. The wafer itself has to take over the part of a PCB. All required nets are routed in three layers on the wafer. The physical realization is done in the HLL.

PCB design tools are not sufficient for routing on wafer level. Therefore, an ASIC design tool from the company Cadence is used. This tool provides the possibility to describe and design semiconductor logic for electrical components. These structures are processed on a wafer, usually cut out and put into protective cases. Just the pins stick out and provide access to the ports of the electrical logic. However, it provides the possibility to rout the nets of the PXD modules as well. A problem arises for the netlist export in respect to boundary-scan.

A JTAG software expects a netlist created by a PCB design tool and describing a PCB circuitry. The problematic differences are described below.

5.1. EDIF Format

The netlist file for the PXD modules is generated automatically by the design and layout software. The only format provided is the Electronic Design Interchange Format (EDIF) in version 2 0 0. EDIF was one of the first attempts to define a neutral data exchange format for electronic designs. However, the rules were not specific enough. With EDIF 2 0 0 it is possible to describe the same data in several ways. This led to the so called "flavors" of EDIF [26]. As a result, software has often problems to import EDIF files and companies providing JTAG tools recommend to not use EDIF.

5.1.1. Keyword Cell

The software used for the routing on wafer level assumes that a large IC is described. A component of the circuitry is explained as `cell`, like a part of an electrical logic. A `cell` can have several input ports and output ports for receiving and transmitting data. The ports are identified by their logical port names, like in the BSDL files. But unlike packaged ICs, the `cells` do not have physical pins. In a PCB design software, every pin of a logical components is described by the logical port name and the physical pin number. Therefore, the pin numbers are usually stated in the netlist. A JTAG software reading a netlist file tries of find matching pin numbers in the `PIN_MAP` section of the corresponding BSDL file. This is not possible for the netlist of the PXD modules.

To fix this, a python script was written after the alphanumerical pin values were considered in the BSDL files. The python script reads the BSDL files and stores the assignment of logical port names to the corresponding pin number. Thereafter, the script tries to find matching logical port names in the netlist file and replaces them by the stored pin number. Some manual editing was necessary as well because sometimes different logical port names for the identical port were used in the manual and in the layout.

```
(cell dhp10_Footprint (cellType GENERIC)
   (view symbol (viewType NETLIST)
    (interface
     (port (rename BitClkDiffN "E36") (direction OUTPUT))
     (port (rename VSS "A16") (direction INPUT))
     (port (rename DES_CLK_P "F4") (direction INPUT))
     .
     (port (rename DI1_3 "B26") (direction INPUT))
     .
     .
```

```
    (contents
      (net (rename DI2_7 "DI2<7>") (joined
 (portRef  DI2_7)))
      (net (rename DI2_6 "DI2<6>") (joined
 (portRef  DI2_6)))
 .
 .
 .
      (net (rename DI1_0 "DI1<0>") (joined
 (portRef  DI1_0))))))
```

Under the keyword `interface`, all ports of a component are listed. This is the `interface` which can be used to communicate with this `cell`. Beneath the keyword `contents` all parts of this `cell` are listed. However, for the PXD modules the `cells` are just the footprints of the ASICs and consist therefore only of the contact pads (`nets`) for the solder balls.

5.1.2. LVS_E_Module3

The description of the interconnections between the individual `cells` are listed in the `cell` `LVS_E_Module3`. It represents the whole all-silicon module. It is also defined as `cell` but it has no `ports` defined for an `interface`. All components like resistors, bondpads, ASICs and the matrix, which were defined as `cells` before, build the `contents`. They are listed as `instances`.

```
  (cell LVS_E_Module3 (cellType GENERIC)
   (view schematic (viewType NETLIST)
    (interface)
(property (rename instance_35_ "instance#") (integer 339) (owner "Cadence"))
(property (rename pin_35_ "pin#") (integer 10) (owner "Cadence"))
.
.
.
    (contents
      (instance R10
       (viewRef symbol (cellRef smd_res_01005))
(property r (string "100")))
      (instance I127
       (viewRef symbol (cellRef bondpad_300x85)))
      .
      .
      (instance dhp2
       (viewRef symbol (cellRef dhp10_Footprint)))
       .
```

```
       .
       .
     (net (rename net0135_4 "net0135<4>") (joined
(portRef  DO1_3 (instanceRef  dcd3))
(portRef  DI1_3 (instanceRef  dhp3))))
.
.
.
```

5.1.3. Keyword Net

The last section in the `cell LVS_E_Module3` describes which nets it contains. They correspond to the routing of the PXD modules. In the example above, the `net0135<4>` is shown. It connects port `DO1_3` of `dcd3` to port `DI1_3` of `dhp3`. Due to the renaming in the `cell` definitions of the ASICs, the JTAG software can assign the pin numbers and get the required information: Pin "P4" of DCD3 should be connected to pin "B26" of DHP3.

5.2. Fully Populated Modules

One special fact about the routing of the PXD modules has to be considered in addition. The JTAG signals are usually 3.3 V Low-Voltage-TTL (LVTTL) single ended signals. But the ASICs are built in 1.8 V CMOS technology and the DHPs communicate with the outer world through LVDS. The first ASIC in the JTAG chain is therefore always a DHP. It receives all data through LVDS and converts the signals into single ended signals for the DCDs and Switchers. It is also possible to dynamically configure the JTAG chain. Every DHP has two possible TDO pins. One is connected to the neighboring DCD, the other one to the next DHP (or to the TDO output of the module for the last DHP). By setting the corresponding configuration in the internal register of a DHP, it is possible to switch between the two TDO outputs (see Figure 5.1).

The shortest possible JTAG chain for a fully populated module contains only the four DHPs. It is possible to expand the chain after each DHP by one DCD. At the fourth DHP, all six Switchers are automatically integrated in the chain as well. A DHP has only one differential TDI input and a DCD (or Switcher) only a single ended TDO, so it is necessary to guide the signal from the DCDs (or the last Switcher) back to the DHP, where the chain was extended. Then the signals are converted into LVDS and transferred to the next DHP (or to the TDO of the module).

This special configuration leads to a discrepancy between signal flow and order in the JTAG chain. A DHP is not integrated twice into the JTAG chain when the TDO signal of the DCD (or Switcher) is only converted and sent to the next component. Furthermore, the DHP input pin for the DCD's (or Switchers) TDO signal is not stated to be a TDI pin

(and it is not a TDI pin). In fact, there is only one TDI pin definition possible in a BSDL file. But if the TDO pin of the DCD (or Switcher) is not connected to a valid TDI pin, the JTAG chain is broken. It was necessary to hide the actual routing at that point in the netlist and to claim a direct connection from the TDO of th DCD (or Switcher) to the next TDI. Otherwise, the JTAG software assumes that the DHPs should be integrated in the JTAG chain a second time.

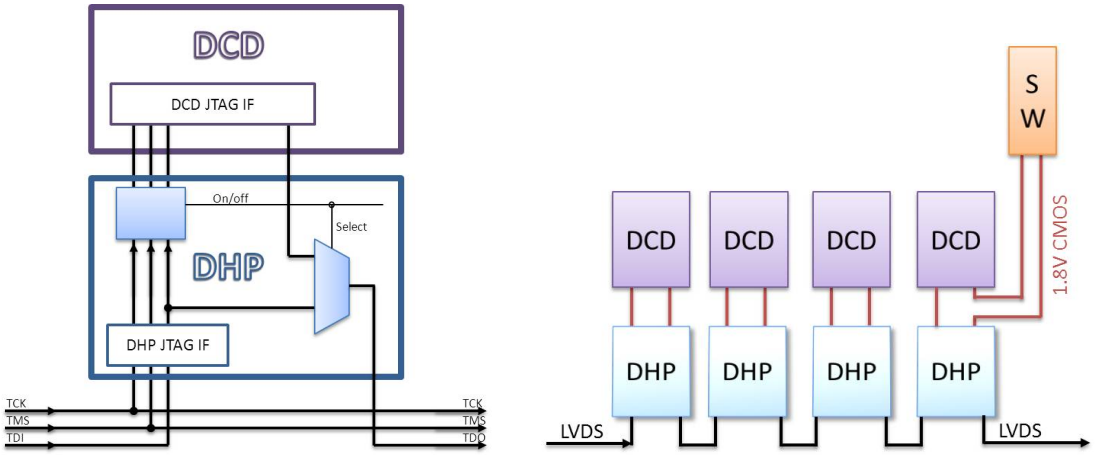


Figure 5.1.: Figure (a) shows the JTAG signal flow from the DHP to the DCD. In Figure (b) the whole chain on a fully populated module is depicted. The black lines are mandatory, the brown lines can be switched on and off in the DHPs. [22]

5.3. Partially Populated Modules

Not every module is fully equipped with all ASICs because it is not necessary for the testing and debugging of special properties. These partially populated modules have the same physical routing. Only some ASICs are missing and therefore some pads are not occupied. However, a closed JTAG chain through all ASICs must be provided. A different netlist file is not necessary. The JTAG software will find the devices in the netlist, although they are not mounted. But the BSDL files can be adapted in a way that these missing ASICs are no longer part of the JTAG chain (see subsection 6.3.2). The software will ignore them.

6

XJTAG System

Several different companies offer boundary-scan solutions in form of software and hardware tools. For the PXD modules a system of the company XJTAG is used. It offers a wide range of different JTAG programs which address specific purposes.

XJDeveloper	Development and debugging of boundary-scan tests Schematic and Layout Viewer Test coverage analysis
XJInvestigator	Real-time, pin-by-pin control of BGA and fine-pitch devices In-system programming
XJAnalyser	Real-time circuit visualisation and debugging Graphical view of JTAG chains
XJRunner	Executing of XJDeveloper tests during production
JTAG Chain Debugger	Simple tool to investigate faulty JTAG chains Check of ID-Codes and maximum chain frequency

[27]

A lot of different hardware is available, too, to fit all possible demands, from little scale up to mass production. Both versions of the smallest JTAG controller (XJLink, XJLink2), which are hand-held devices, are sufficient for the boundary-scan tests during the PXD module production.

6.1. Boundary-scan with XJTAG

The use of the XJTAG system is already intended during the design process of an electronic device. A first design version of a new device can be loaded into the XJDeveloper together with the BSDL files of the ICs, which use is considered. With this information and without the need of the actual device, the XJDeveloper is able to check the design for testability and to calculate the expected test coverage. The recommended design format is ODB++, which usually contains schematic and layout of a design as well. The Schematic and Layout Viewer of the XJTAG software can point on problematic issues directly in these files. The designer can use this information to rework the design to gain higher test coverage. The XJDeveloper is used to create the boundary-scan tests as well. The netlist and the BSDL files provide sufficient information for the software to automatically create an interconnection test. More complex tests, like testing a logic in between boundary-scan ICs, are also possible and can be programmed with XJEase, a flexible high-level test description language.

During the production, the actual boundary-scan tests are performed before the devices undergo the next production step. The XJRunner is provided for this purpose, although it is possible to execute the tests with the XJDeveloper as well. But the XJRunner only allows for the execution of predefined tests without the possibility for configurations. This application is usually used in big company where the tester does not need to have experience in design and test analysis.

XJInvestigator and XJAnalyser are special debugging tools for repair and rework or visualization and debugging respectively. The JTAG Chain Debugger is the basic tool for checking the chain integrity. It can perform *blind interrogation* and check ID-Codes and IR length after the assignment of BSDL files. It can determine the maximum possible JTAG frequency as well.

6.2. XJLink Controller

JTAG software runs on PCs and needs some conversion hardware to provide the JTAG signals. The most common PC interface is USB. Therefore, most JTAG controllers are USB to JTAG converters. There are some special versions for e.g. Ethernet or PXI[1] rack as well [28]. For the boundary-scan tests of the PXD modules, the USB hand-held devices of XJTAG were used.

6.2.1. XJLink

The XJLink is the first version of the small and portable XJTAG controller. It is an USB 2.0 to JTAG adapter with a configurable 20-way connector. The possibility to assign the

[1] PCI (Peripheral Component Interconnect) eXtensions for Instrumentation

JTAG signals freely to the individual pins within the software, accounts for the fact that there is no common standard for JTAG headers. It is only able to control one TAP and therefore only one JTAG chain at a time. The maximum TCK frequency is 50 MHz. The XJLink contains the license keys for the XJTAG software as well. The whole XJTAG software can be installed on as many PCs as necessary but to run the individual program, a so called dongle with valid licenses must be connected to the PC. The license management, i.e. updating an old license with a new key, is done in a small program called "XJLink Manager".

Figure 6.1.: XJLink Controller.

6.2.2. XJLink2

The XJLink2 is the successor of the XJLink. It has the same configuration but also some improvements. It can handle up to four TAP controllers. The maximum TCK frequency is 150 MHz. It has a press button to start a loaded boundary-scan test from the hardware and three LEDs to indicate the test status as well.

The different supply power demands were the only significant issue which had influence on the testing. To operate the XJLink on a Laptop USB port, an externally powered USB Hub was required. This was not necessary for the XJLink2.

Figure 6.2.: XJLink2 Controller. [28]

6.3. PXD Module Project

The boundary-scan tests for the PXD modules were created and executed with the XJDeveloper 3.3 software. All test and configuration files of the XJTAG software for a specific device are summarized under the term "project".

Two special parameters must be set before the import of the EDIF netlist because it is important to state which `cell` describes the whole module. This information was given after contacting the XJTAG support.

Top Cell:	LVS_E_Module3
Top Lib:	E_Module3

6.3.1. Definition of Components

The next step, after the import of the netlist file, is the definition of the components. All components are listed as `instances` in the EDIF netlist. The names are arbitrary and every component must be defined for the software so that it knows how to handle them. At first, the JTAG components are assigned. With the correct identification of all components, XJTAG is able to test pull-up and pull-down resistors as well. The software suggests that

all components which names begin with "R" are resistors. But it has to be specified which of them are for example termination resistors for LVDS lines and can therefore not be treated like pull-up or pull-down resistors. The kapton footprint on the module is defined as connector.

6.3.2. JTAG Chain

After all ASICs are declared as JTAG devices, the starting point of the JTAG chain must be defined. The TDI signal comes from the kapton cable. The first element of the TDI line is therefore the corresponding bond pad on the PXD module. The bond pads are described in the `cell kapton_Footprint_14mm` in the netlist. There are two matching pads: `port TDI_P` and `port TDI_SINGLE_P`. Which one has to be used depends on the population of the module under test. The `port TDI_P` is connected to the first DHP and has to be chosen for the fully populated version. The `port TDI_SINGLE_P` is connected to the fourth DHP and has to be chosen for a partially populated module with only one DHP. However, the TDO pad is the same for both versions: `port TDO_P`.

When the starting point of the JTAG chain is declared, the XJDeveloper tries to find the succeeding parts. The first DHP is easily found but right after it are two possibilities how the chain can proceed. As the IEEE Std 1149.1 allows only for one TDO, the software searches for the corresponding description in the BSDL file of the DHP. In the default `version 1`, the TDO output to the neighboring DCD is used. If, for some reason, the JTAG chain is bypassing the corresponding DCD, the entry in the BSDL file has to be changed to `version 2`. This can be done directly in the XJDeveloper software with the integrated editor.

```
attribute TAP_SCAN_OUT   of DCD_TDO   : signal is true;  --  version 1

attribute TAP_SCAN_OUT   of TDO_P     : signal is true;  --  version 2
```

If `version 2` is used, the software accepts the second DHP as next device in the chain and the same procedure can be repeated for the TDO output of the second DHP. If `version 1` is chosen, the XJDeveloper takes the DCD as second device in the chain but it gets confused about the fact that the TDO of the DCD enters again the first DHP. This problem was solved by changing the netlist file at that point (see section 5.2). The actual netlist file can be universally used, independent of the present population of the module. The different versions have to be considered by changing the TDI and TDO definitions in the BSDL files properly.

The JTAG chain integrity test and the interconnection test are automatically generated by the XJTAG software. Both are added to the test section of the "PXD Module Project". The default settings in the BSDL files consider a full JTAG chain, as it is expected (and required) for the series production modules. The JTAG software is prepared and the boundary-scan tests can be executed after the connection of the modules only by clicking the "RUN" button.

6.3.3. Test Coverage

The test coverage can be defined in several ways. It is possible to look at the nets of a boards or at the pins of the components. For the latter, the frame of reference can be all pins on the board including the pins of all passive components, like resistors and connectors, or just the pins of the ICs. The two possible pin test coverages are discussed in this section.

The XJDeveloper calculates the test coverage referring to all pins on a board. After the description of the PXD module project, a test coverage of ~19.4 % is claimed (Figure 6.3). The power pins are listed with a share of ~8.2 %. They can not be tested individually. However, it is common that several pins are used for the power supply of an IC to assure a stable operation. In the BGA of the ASICs, the power pins are grouped together. A short can only occur between neighboring solder balls and the ASICs will work properly as long as both belong to the same net. Therefore, if a boundary-scan test passes, it is not clear whether all of the solder balls at the power pins are fine. But this is not important, as long as the system is working correctly. In combination, this results in a pin test coverage, in respect to all pins on the module of ~27.6 %.

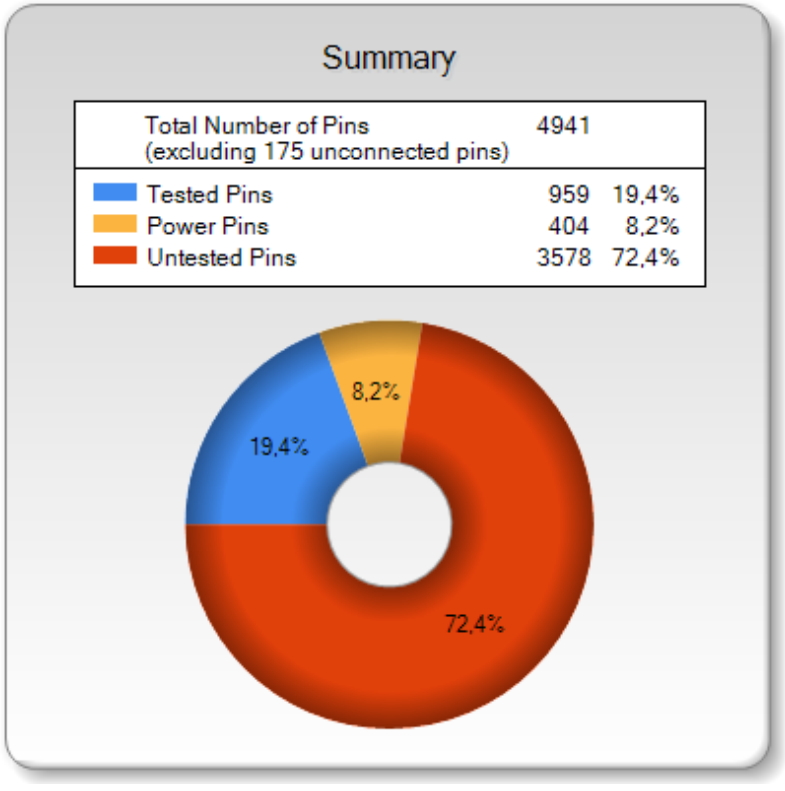


Figure 6.3.: Screenshot of the calculated pin test coverage of the XJDeveloper.

The pin test coverage, in reference to the total number of all ASIC pins, can easily be calculated. The following table lists the total number of active pins of the ASICs as well as all pins which are directly tested by a boundary-scan test.

	active pins	directly tested pins	sum
Switcher	96	BSCs = 5 (all LVDS) = 10 pins	14
		JTAG → +4 pins	
DCD	431	BSCs = 83 pins	88
		JTAG → +4 pins	
		RefOut → +1 pin	
DHPT1.0	255	BSCs = 115 − 24 (control cells) = 91 pins	111
		7 BSCs are LVDS → +7 pins	
		LVDS JTAG → +8 pins	
		4 JTAG signals for DCD → +4 pins	
		RefIn → +1 pin	

For the DHP, the newer version is chosen. The different versions of the other ASIC types have the same values.

On one PXD module, six Switchers, four DCDs and four DHPs are mounted. Their active pins sum up to a total number of 3320 per module of which 880 can be directly tested with a boundary-scan test. This leads to a direct pin test coverage in respect to the total number of active ASIC pins of

$$\frac{880}{3320} \approx 26.5\,\%$$

These numbers are not very high. Over 70 % of the pins remain untested. But the PXD is still an analog sensor and most of the pins are therefore analog I/Os. However, a faulty solder connection on an analog pin results in several dead pixels which can not be used. This decreases the performance of the detector but it is (within a certain range) not a severe problem, which hinders the operation of the PXD. Defects in the digital part of a module, e.g. a broken JTAG line, can make it impossible to use it for the final detector. With the use of boundary-scan during the module production, faults in the digital part can easily be found. After the analysis of the faults, it is possible to point out where they occurred and if it is feasible to rework the module as well.

7

JTAG Breakout Board

The DHPs communicate with the controlling DHE through LVDS lines. The design of the DHPT0.2 requires 14 data lines between a module and a DHE for this interface. The DHPT1.0 design enabled a reduction of this number by time-multiplexing several signals into one line to the following twelve lines:

JTAG signals	TCK	
	TMS	
	TDI	
	TDO	
high speed data links	TX_0	
	TX_1	
	TX_2	
	TX_3	
timing signals	GCK	Global Reference Clock
	TRG	four time-multiplexed signals: reset, veto, trigger and frame sync
measurement lines	DHP_SENSE	
	DCD_CURMON	

For JTAG boundary-scans, the four TAP signals as well as the DHP_SENSE line are important. The DHP_SENSE line has a positive and negative line although it is not a LVDS line. The positive line is connected to the supply voltage of the DHPs (VDDD) and the

negative line to the digital ground (GNDD). It results in a voltage difference of 1.8 V on these lines when the DHPs are powered. As long as the DHPs are switched off, no signals should be sent to their inputs so that they are not harm. The DHP_SENSE line is therefore used to switch the DHE outputs to the module on and off.

The two facts that the TAP signals have to be provided with LVDS instead of LVTTL and that the DHP_SENSE needs to be considered made it necessary, to use an additional converter board for boundary-scan tests.

7.1. Sketch

The first sketch of the converter board was provided by the DHE team from the TU Munich, Igor Konorov and Dmytro Levit. An edited sketch of the actual design is shown in Figure 7.1.

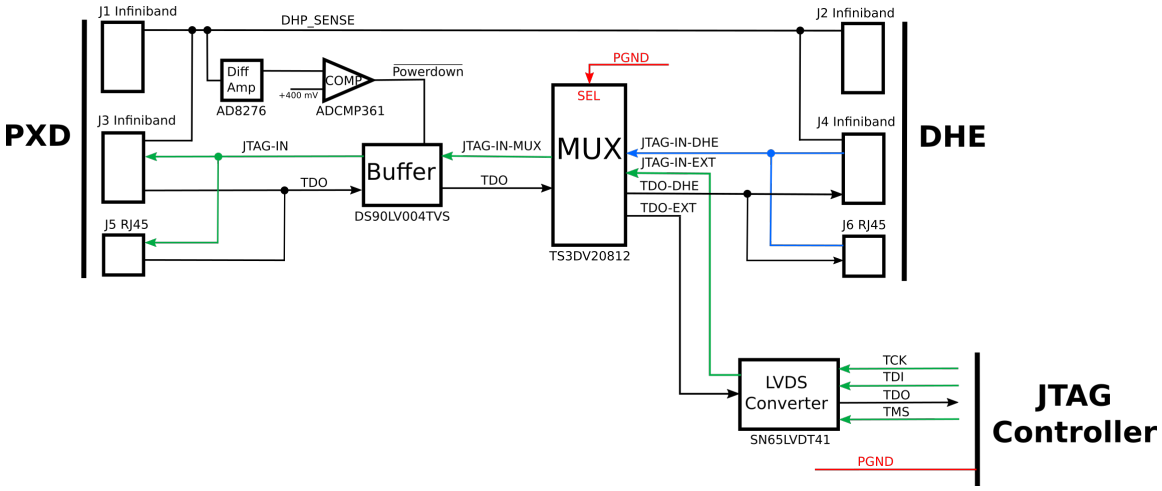


Figure 7.1.: Sketch of the JBB with switched multiplexer.

The already mentioned reduction of the necessary line number between PXD modules and DHE made it possible to replace one Infiniband cable by a RJ45 cable, which provides four shielded twisted pair copper cables instead of the previous eight. This change affected the DHE as well as the patch panel. Therefore, in both new designs the second Infiniband socket is replaced by a RJ45 socket. However, the JTAG converter board was designed to be compatible with both versions. It allows for cross versions between old and new designs as well.

Another requirement was the be possible to switch fast and easily between JTAG controller and DHE signals because the ASICs must be configured before and sometimes in between the tests. Therefore, the connection between module and DHE is the default configuration and only during boundary-scan tests the lines are switched to the JTAG controller. Because of its position between module and DHE, the converter board is called JTAG Breakout Board (JBB).

The JTAG and the High Speed (HS) data lines are never in the same cable, for the old connector version as well as for the new one. Therefore, the cables are also labeled JTAG and HS. Concerning the old version, it is sufficient to use only the lines from the JTAG cable because the DHP_SENSE line could be placed in the Infiniband cable as well. As the new JTAG cable, the RJ45 cable is used and it occupies all four lines. The DHP_SENSE line was therefore transferred to the HS cable.

The lines which are not used for the boundary-scan tests are routed straight from one side to the other on the top layer of the JBB. The four high speed data links were especially considered to have an impedance of 100 Ω. The width of the routed lines and the spacing in between them was chosen to be 150 µm. In discussion with the manufacturer, the distances between the four layers of the PCB were adapted to reach the required impedance control. However, the two additional connecters certainly decrease the signal quality. So it is not recommended to perform tests of the HS links, as long as the JBB is connected in between module and DHE.

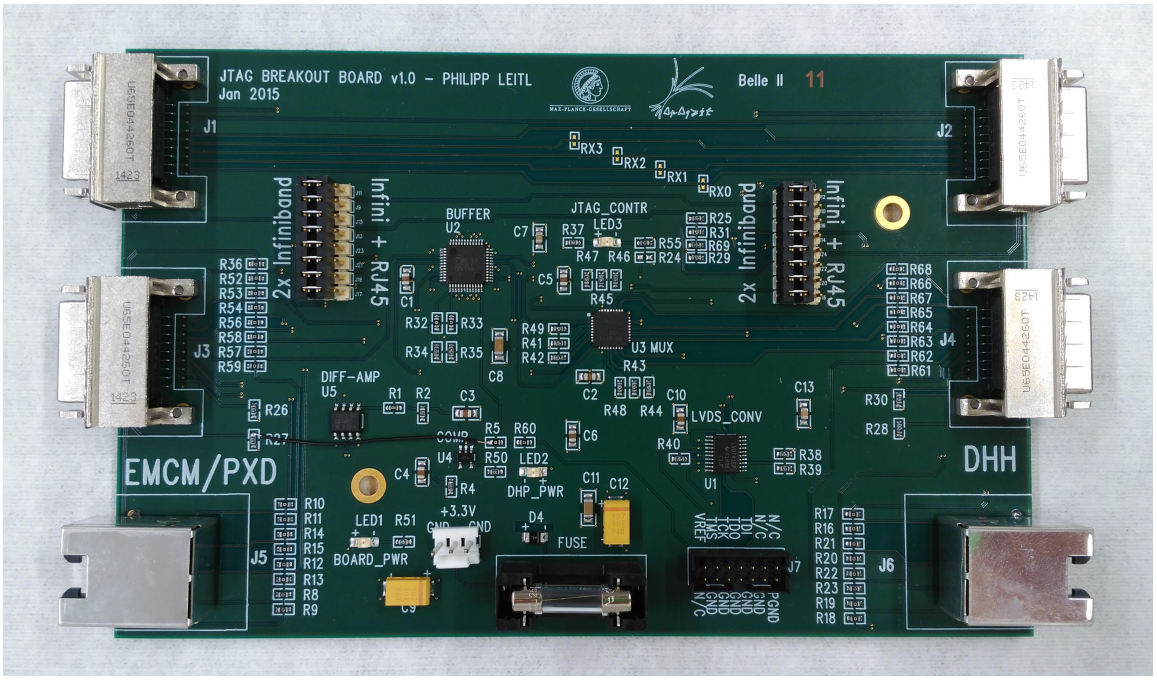


Figure 7.2.: Version 1.0 of the JBB. Shown is board number 11. In total eleven JBB were produced.

7.2. Components

The functionality of the JBB is described by the purposes of its components. The output circuitry to the DHP lines is identical to the corresponding circuitry on the DHE.

7.2.1. Multiplexer

The core of the JBB is the LVDS multiplexer TS3DV20812, where all JTAG data lines come together. The potential at the Select (SEL) input causes the multiplexer to switch between the JTAG lines of the DHE and the ones of the JTAG controller. A pull-up resistor ensures that the default configuration allows the DHE to communicate with the DHPs.

7.2.2. LVDS Converter

The actual task of converting the single ended JTAG lines into LVDS is performed by the converter chip SN65LVDT41. It provides four LVDS line drivers (of with three are used) and one terminated LVDS receiver, which converts back the TDO signal coming from the module into single ended.

7.2.3. Buffer

The multiplexed JTAG signals are fed into the LVDS Buffer DS90LV004. This chip has exactly the required four channels with LVDS transmitters at the outputs. This ensures a good signal quality before the signals are sent further to the module. The chip has the possibility of pre-emphasizing the signals as well. In the default configuration, it is disabled but it can be enabled by desoldering resistors R32 and R33. The Buffer has an inverted power-down pin as well. The potential at this pin is held high and the Buffer is only working when it is pulled down. The DHP_SENSE line can not be used directly to control this power-down pin. The following chips are needed.

7.2.4. Differential Amplifier

The DHP_SENSE consists of two lines which have a voltage difference of 1.8 V as long as the DHPs are powered. The reference ground voltage is not exactly the same for the modules and the DHE / JBB. Therefore, the differential amplifier AD8276 is used to generate an output voltage of 1.8 V compared to the DHE / JBB ground. When the DHPs are not powered, the output of the differential amplifier is 0 V.

7.2.5. Comparator

The output of the differential amplifier is first connected to a voltage divider (R1 and R2). The produced output is then connected to the comparator ADCMP361. It compares the input voltage to an internal reference voltage of 400 mV. When the input voltage is higher than the reference, the output OUT is pulled down. This output is connected to the

inverted power-down pin of the Buffer and activates it, as long as the input voltage is still above the threshold.

During the operation of the DHE, it was recognized that the resistor values of the voltage divider were not optimal. Therefore, the DHE team suggested to use a value of 4 kΩ for R2. This was taken into account during the SMD mounting of the JBB.

During the designing process of the JBB, it was overlooked that the outputs of the comparator are so called open drain-outputs. This means that they do not change between high and low but between open-drain and low. Therefore, the comparator is not able to pull-up the output line. A dedicated pull-up resistor is needed on the inverted power-down pin of the buffer to disable it by default. This bug was quickly found by testing the first assembled JBB and fixed by a wire from resistor R5 to resistor R27.

7.2.6. Connectors

The Infiniband (U65-E04-4260T) and RJ45 (SS-60000-009) connectors were deliberately chosen to have stable positioning pins. Especially the pins of the Infiniband connector, which are SMD soldered onto the PCB, are sensitive. The positioning pins protect them during the frequent connecting and disconnecting of the cables.

7.3. Schematic

The schematic was drawn with the software PADS Logic, using the sketch as template. The two sides are shown on pages 69 and 70.

It is necessary to switch also the lines on the board when a different connecter version is used. This is done by jumpers. When one Infiniband cable is replaced by a RJ45 cable, eight lines have to be switched. This results in 16 jumpers for the whole JBB.

7.4. Layout

The layout was created with the software PADS Layout. It is shown on page 71.

The connectors are placed on two sides. On one side there are the connectors for the module (labeled EMCM/PXD) and on the other one are the connectors for the DHE (labeled DHH). The uppermost connectors belong to the HS cables, which have to be used in all version. The ones in the middle are for the optional Infiniband cables of the old version and the remaining ones are for the RJ45 cables.

The JTAG connector is the 14-pin header next to the RJ45 connector on the right side. The signal names are label because there is no common standard about the JTAG pin configuration.

The 16 jumpers are arranged in two columns of eight jumpers each. Every column belongs to one side. All jumpers of a column are switched at once. The eight-pin headers can be placed on the left or on the right side of each column. The corresponding labels indicate which jumper position is required for which cable configuration. The logic and layout was designed in a way that also the cross versions are possible, e.g. an old module with a new DHE.

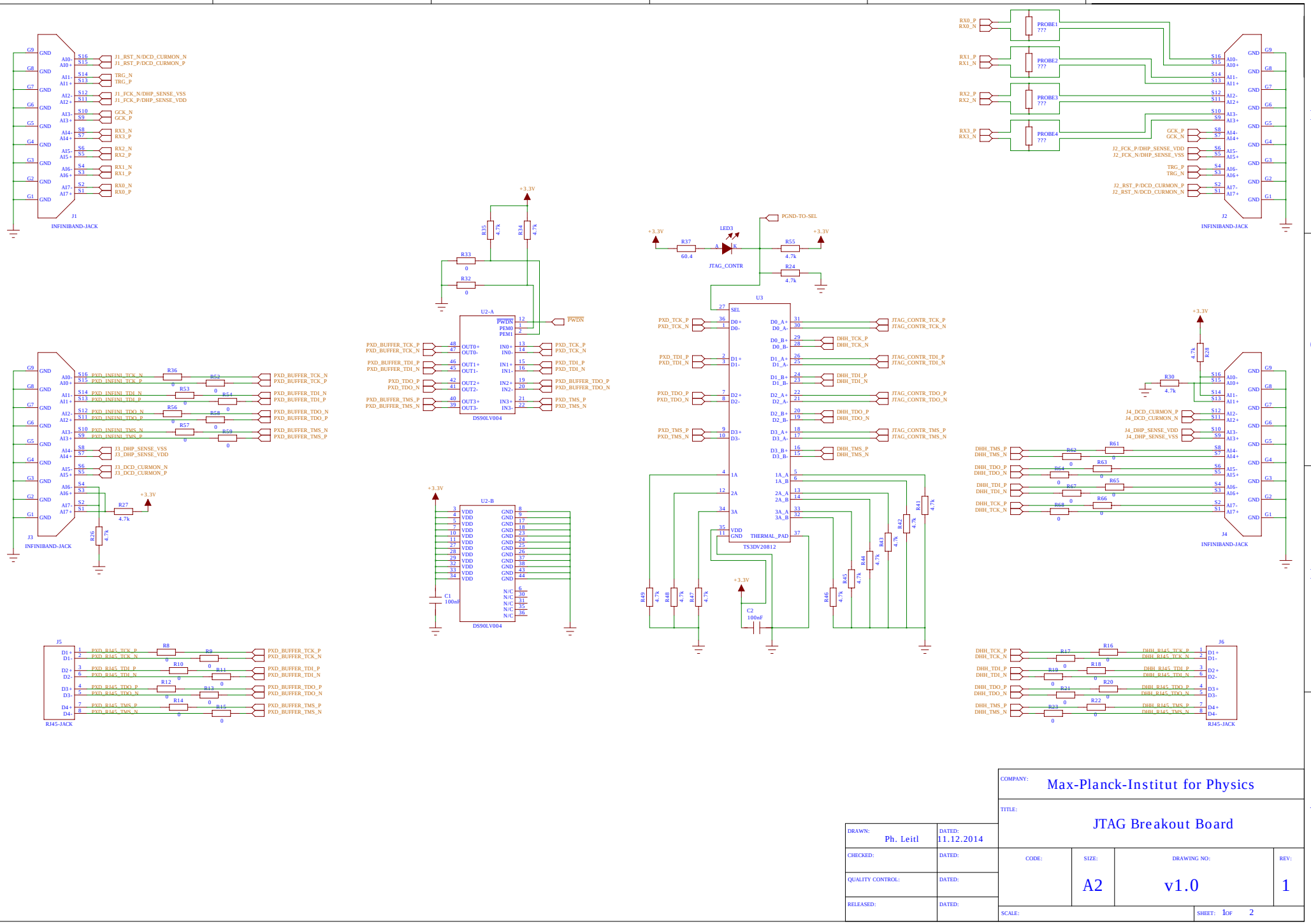
7.5. Operating Instructions

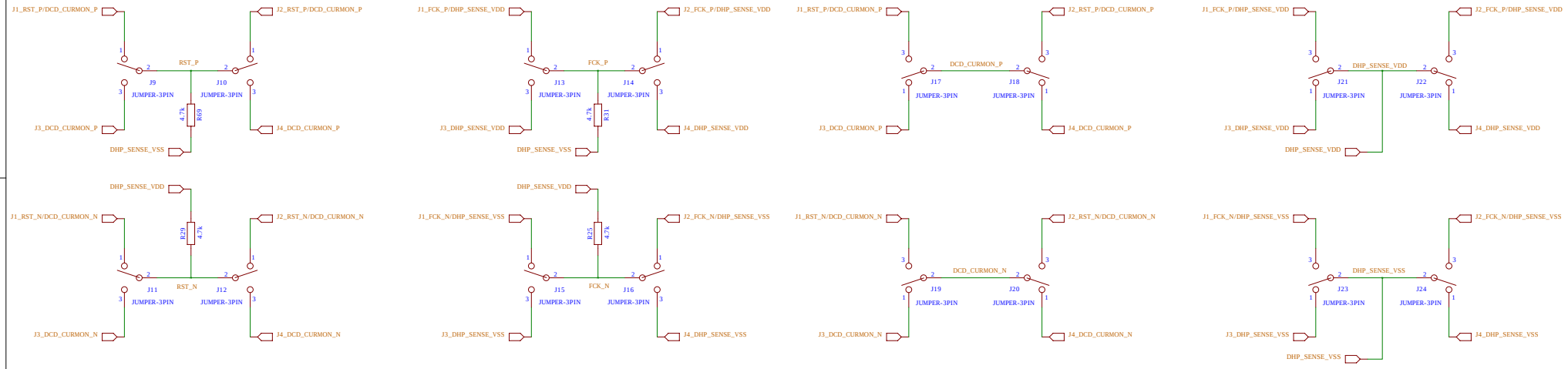
The JBB needs its own power supply, which has to be connected to the 3-pin power connector. The supply voltage is 3.3 V and for the current limit 300 mA are recommended. The +3.3 V pin is the middle, the two pins on the side are for ground, as labeled.

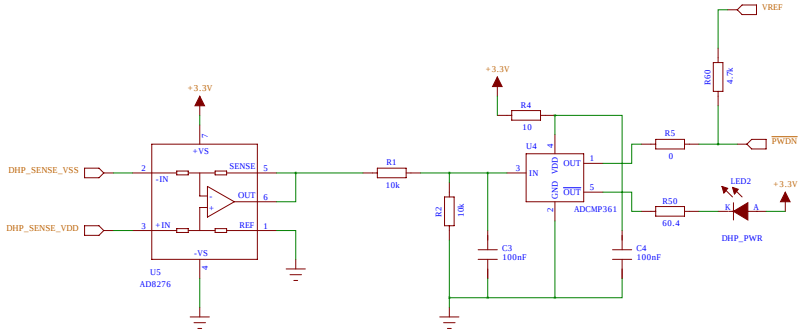
The power should be switched off during connecting the cables. Before powering, all connections and the position of the jumpers have to be checked. As long as the JBB is in between the module and the DHE and as long as it is not powered, no signals can be transmitted. When the JBB is powered, the LED1 is switched on. LED2 indicates with a green light when the DHPs are powered and the outputs are active.

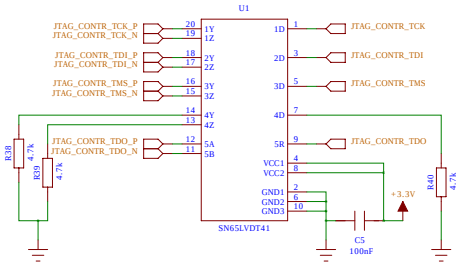
The "PXD Module Project" of the XJDeveloper software instructs the XJLink to pull down the SEL line of the multiplexer before the test signals are sent. After a boundary-scan test, this pin of the XJLink is switched to high impedance, and the pull-up resistor on the SEL line switches back the multiplexer to the DHE signals. As long as the JTAG controller is performing, the test LED3 shines green as well.

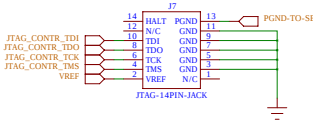
The documentation and operating instructions can be found on the following TWiki page: http://twiki.hll.mpg.de/bin/view/DepfetInternal/JTAG20Breakout20Board20(JBB)

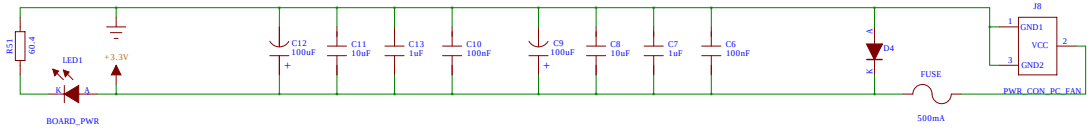

J1
INFINIBAND-JACK
J1_RST_N/DCD_CURMON_N
J1_RST_P/DCD_CURMON_P
TRG_N
TRG_P
J1_FCK_N/DHP_SENSE_VSS
J1_FCK_P/DHP_SENSE_VDD
GCK_N
GCK_P
RX3_N
RX3_P
RX2_N
RX2_P
RX1_N
RX1_P
RX0_N
RX0_P
PROBE1
PROBE2
PROBE3
PROBE4
J2
INFINIBAND-JACK
J2_FCK_P/DHP_SENSE_VDD
J2_FCK_N/DHP_SENSE_VSS
J2_RST_P/DCD_CURMON_P
J2_RST_N/DCD_CURMON_N
J3
INFINIBAND-JACK
PXD_INFINI_TCK_N
PXD_INFINI_TCK_P
PXD_INFINI_TDI_N
PXD_INFINI_TDI_P
PXD_INFINI_TDO_N
PXD_INFINI_TDO_P
PXD_INFINI_TMS_N
PXD_INFINI_TMS_P
J3_DHP_SENSE_VSS
J3_DHP_SENSE_VDD
J3_DCD_CURMON_N
J3_DCD_CURMON_P
J4
INFINIBAND-JACK
J4_DCD_CURMON_P
J4_DCD_CURMON_N
J4_DHP_SENSE_VDD
J4_DHP_SENSE_VSS
J5
RJ45-JACK
PXD_RJ45_TCK_P
PXD_RJ45_TCK_N
PXD_RJ45_TDI_P
PXD_RJ45_TDI_N
PXD_RJ45_TDO_P
PXD_RJ45_TDO_N
PXD_RJ45_TMS_P
PXD_RJ45_TMS_N
J6
RJ45-JACK
DHH_RJ45_TCK_P
DHH_RJ45_TCK_N
DHH_RJ45_TDI_P
DHH_RJ45_TDI_N
DHH_RJ45_TDO_P
DHH_RJ45_TDO_N
DHH_RJ45_TMS_P
DHH_RJ45_TMS_N
U2-A
U2-B
DS90LV004
PWDN
PXD_BUFFER_TCK_P
PXD_BUFFER_TCK_N
PXD_BUFFER_TDI_P
PXD_BUFFER_TDI_N
PXD_BUFFER_TDO_P
PXD_BUFFER_TDO_N
PXD_BUFFER_TMS_P
PXD_BUFFER_TMS_N
PXD_TCK_P
PXD_TCK_N
PXD_TDI_P
PXD_TDI_N
PXD_TDO_P
PXD_TDO_N
PXD_TMS_P
PXD_TMS_N
U3
TS3DV20812
THERMAL_PAD
JTAG_CONTR
LED3
PGND-TO-SEL
DHH_TCK_P
DHH_TCK_N
DHH_TDI_P
DHH_TDI_N
DHH_TDO_P
DHH_TDO_N
DHH_TMS_P
DHH_TMS_N
JTAG_CONTR_TCK_P
JTAG_CONTR_TCK_N
JTAG_CONTR_TDI_P
JTAG_CONTR_TDI_N
JTAG_CONTR_TDO_P
JTAG_CONTR_TDO_N
JTAG_CONTR_TMS_P
JTAG_CONTR_TMS_N
+3.3V
C1
100nF
C2
100nF
COMPANY:
Max-Planck-Institut for Physics
TITLE:
JTAG Breakout Board
DRAWN:
Ph. Leitl
DATED:
11.12.2014
CHECKED:
DATED:
QUALITY CONTROL:
DATED:
RELEASED:
DATED:
CODE:
SIZE:
A2
DRAWING NO:
v1.0
REV:
1
SCALE:
SHEET:
1
OF
2



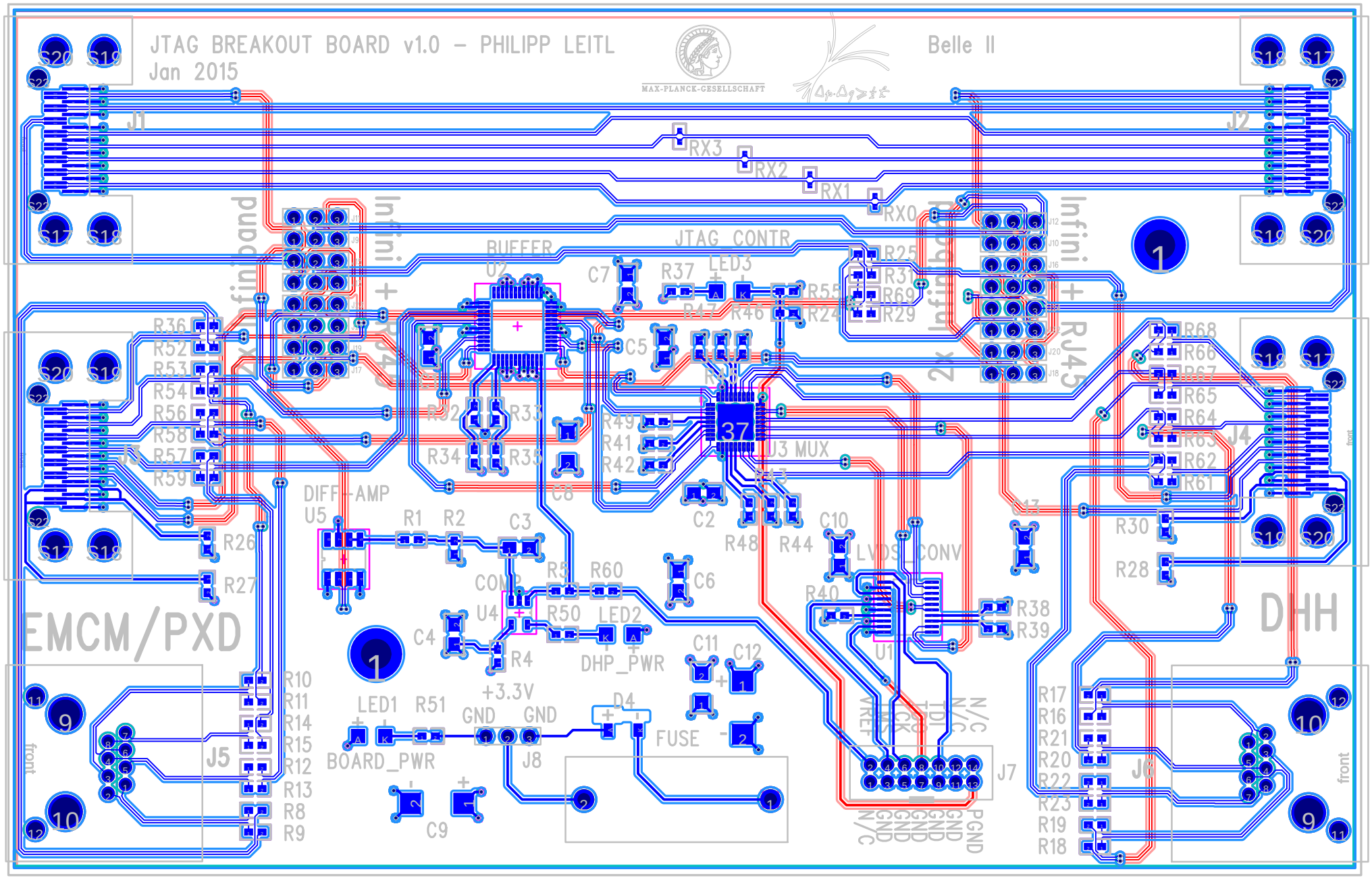








COMPANY:	Max-Planck-Institut for Physics		
TITLE:	JTAG Breakout Board		
DRAWN: Ph. Leitl	DATED: 11.12.2014	CODE:	SIZE: A2 / DRAWING NO: v1.0 / REV: 1
CHECKED:	DATED:		
QUALITY CONTROL:	DATED:		
RELEASED:	DATED:	SCALE:	SHEET: 2 OF 2


JTAG BREAKOUT BOARD v1.0 – PHILIPP LEITL
Jan 2015
MAX-PLANCK-GESELLSCHAFT
Belle II
EMCM/PXD
DHH
J1
J2
J3
J4
J5
J6
J7
J8
U1
U2
U3 MUX
U4
U5
DIFF-AMP
BUFFER
JTAG CONTR
LVDS CONV
COMP
LED1
LED2
LED3
BOARD_PWR
DHP_PWR
FUSE
+3.3V
GND
2x Infiniband + RJ45
2x Infiniband + RJ45
front
front

Part III.

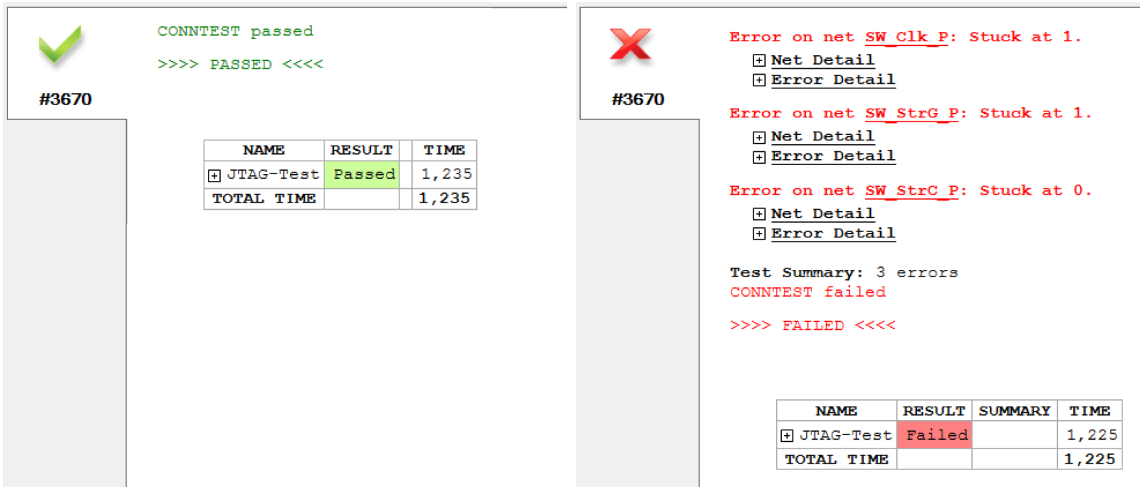
Boundary-Scan Measurements and Results

8

Boundary-scan Test of EMCM and PXD9 Modules

Several measurements were necessary to debug the whole boundary-scan system consisting of self-developed hardware (JBB), proprietary JTAG tool (XJTAG) and customized input files (BSDL and netlist). During this process, eight different modules were tested, of which six showed no faults in the boundary-scan tests. The two other ones and their faults are described in more detail below.

"No faults" means in this context that the chain integrity check and the interconnection test passed without a single error. This result confirms that the JTAG communication works without restrictions and that all nets connected to the covered pins (see subsection 6.3.3) work as intended. These modules are called "golden". The DHP offers a possibility to double check the JTAG system on such a module. The fourth DHP is connected to the switchers and generates control signals for them. These signal lines are tested by the interconnection test as well. The output drivers for these signals can be switched on and off by the right configuration of the DHP. The `sw_en_out` must be enabled in the global register as well as settings for the driver strength `sw_tx_set*`. The JTAG software, however, has no information about the status of the drivers and will always try to drive these lines. Therefore, it is possible to generate a fault on a "golden" module by disabling these output drivers. The described JTAG system was able to detect this artificial fault on all modules. An exemplary graphical output of the XJDeveloper for both cases is shown in Figure 8.1.



(a) (b)

Figure 8.1.: Measurement results for an artificial created stuck on Switcher lines. Figure (a) shows the software output for a "golden" module with no faults. Figure (b) shows the result for the same module but this time with disabled output drivers for the Switcher lines.

8.1. Electrical Multi-Chip Modules

The Electrical Multi-Chip Modules (EMCMs) were designed and built as fully functional PXD modules, but without a DEPFET matrix (see Figure 8.2). This was done to check the three-layer metal routing and to practice the mounting of ASICs and passive components as well as the attachment of the kapton cables. Furthermore, the working EMCMs serve as testing environment for ASIC functionality and communication issues. It is possible to attach a small sensor matrix as well.

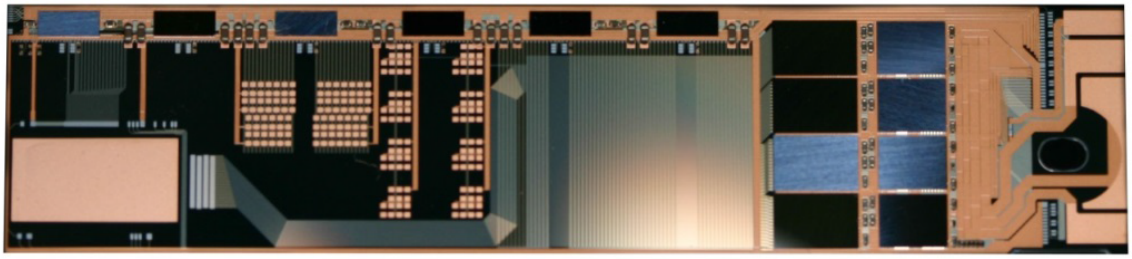


Figure 8.2.: Fully-populated EMCM without matrix.

Seven of the eight tested modules are EMCMs. Two of them (P6-1 and P6-2) are only partially populated with one DHP, one DCD and one Switcher. The other five modules are fully populated with four DHPs, four DCDs and six Switchers corresponding to a total number of 14 ASICs in the JTAG chain. The five EMCMs which passed the boundary-scan test are listed below.

EMCM P6-1 ✓

EMCM P6-2 ✓

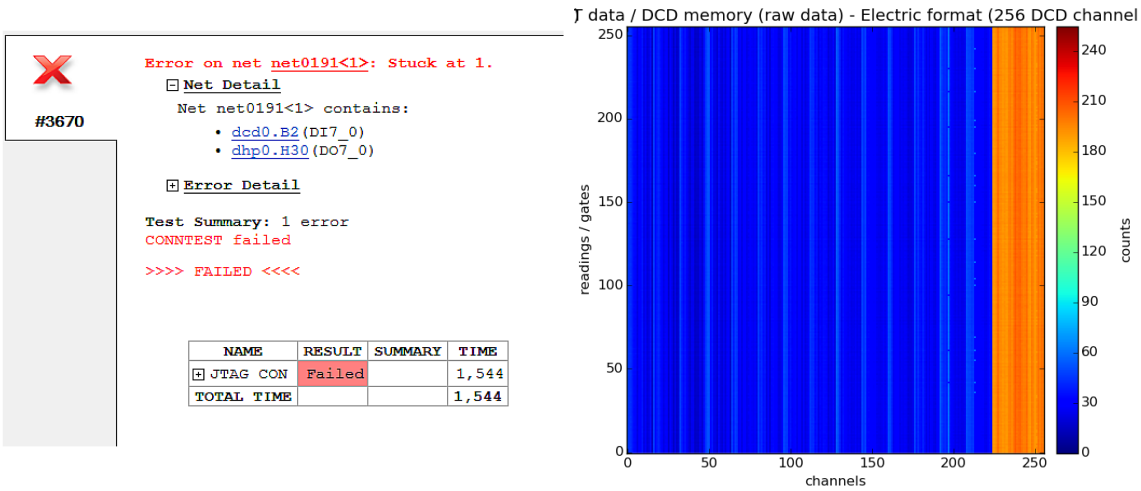
EMCM W18-4 ✓

EMCM W31-3 ✓

EMCM W31-4 ✓

8.1.1. EMCM W18-3

The JTAG chain of the EMCM W18-3 passed the chain integrity check, but the boundary-scan test found one faulty line between the first ASIC pair on this module. The output of the software is shown in Figure 8.3 (a). It names the `net0191<1>` and reports a stuck at 1. It also lists the pins connected to this net. The corresponding port names are given in brackets: Pin "H30" (`DO7_0`) of the DHP and pin "B2" (`DI7_0`) of the DCD. This result indicates that the software tried to write a changing pattern of 1 and 0 from the DHP output. At the connected DCD input it could only read back a 1 all the time. For the data transmission, the DCD is divided into eight columns. Every column has two lines from the DHP to the DCD for the pedestal correction of the pixel values. The affected line is one of the two lines of the eighth column (starting counting at 0). As the communication of the pedestal values of this line is affected, the readout of DCD data shows the expected pattern of corrupted values for the eighth column (see Figure 8.3 (b)).



(a) **(b)**

Figure 8.3.: Figure (a) shows the boundary-scan test result of the EMCM W18-3 in the XJTAG software. Figure (b) shows the raw data readout of the DCD memory for the first ASIC pair on EMCM W18-3.

8.1.2. EMCM W17-4

All tested modules but the EMCM W17-4 have a kapton cable attached. It is straight forward to insert the JBB between patch panel and DHE for these modules. However, boundary-scan is intended to be used during the series production before the attachment of the kapton cable. A probe card is used to contact the pads of a module, which are later wire-bonded and soldered to the kapton. The 73 wire-bond pads are small and must be contacted with needles. Therefore, the probe card is also called needle card. Besides the contacting of the module, the probe card takes over the part of the patch panel as well and provides the connectors for data and power cables. In this configuration, the JBB is inserted between probe card and DHE.

A probe card setup for the PXD modules is provided at the HLL. It was used to test the boundary-scan system in this configuration as well. The critical part is the contact of the needles to the bond pads. They must be pushed down with enough pressure that every needle has a stable contact, but the bond pads must not be damaged so that they can still be used for wire-bonding to the kapton. A microscope is used to control the position of the needles and their bending after touching the surface.

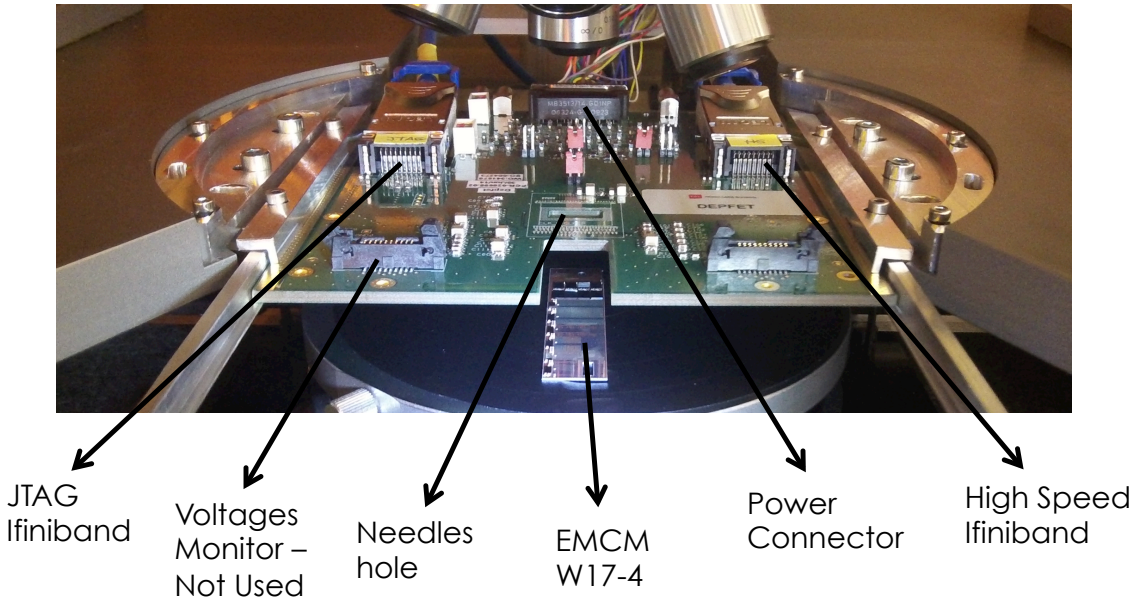


Figure 8.4.: Needle card with EMCM W17-4. [29]

The EMCM W17-4 was used to practice the probe card procedure [29]. The setup was extended with the boundary-scan system and boundary-scan tests of the EMCM were performed. At that time, the XJTAG system was not yet available. The results were achieved with a JTAG tool from goepel electronics (software and JTAG controller) instead, but with the same hardware (JBB) and early versions of the BSDL files.

It was possible to execute the boundary-scan tests with this setup. The chain integrity check passed, but the interconnect test found a fault for the third ASIC pair on this module. An

exemplary part of the output of the goepel software is listed below:

```
.
.
.

- 1- Line NET0141_0 defective:
-73-     1. pin <: OUT DCD2:DO7_7(#P79)  {BScan} DCD_FOOTPRINT      NET0141_0
-73-     2. pin >: In  DHP2:DI7_7(#99)   {BScan} DHP10_FOOTPRINT    NET0141_0
- 8-    Stuck at Low of the line

-24- Test step table of the line NET0141_0:
-25-      Expected                     H L L L H L H L H L L H H L H L H L H H
-28-      Measured <Stuck at low>
-30-     Output pin  DCD2:DO7_7(#P79)  H L L L H L H L H L L H H L H L H L H H
-31-      Input pin   DHP2:DI7_7(#99) >L L L L>L L>L L>L L L>L>L L>L L>L L>L>L

.
.
.
```

The NET0141_0 is stated to be "defective". It connects a digital output pin of the third DCD to an input pin of the corresponding DHP but the verification of this connection failed. The single steps of the test are listed as well. The software tried to drive the output pin with a pattern of changing high and lows, but the value at the input pin stayed at low. This caused the software to diagnose the claimed "Stuck at Low of the line". The same fault is shown for all other 63 digital output lines of this DCD. Again, the result could be verified by a functional test. It was possible to readout the DHP data lines with the probe card as well. The pedestal values which the DHP received from the DCD and sent further to the DHE are shown in Figure 8.5.

Instead of the expected distribution, only one value is shown for all pixels. The value of 128 corresponds exactly to the logical "0" on these lines. The digital communication between DCD and DHP is realized by LVDS as well, although there is just one line for each signal routed. The corresponding second potential for all lines is provided from the DCD at the pin RefOut and detected from the DHP at pin DCD_VREF. The fact that all digital lines from the DCD to the DHP are affected, is firmly pointing to a problem at the reference voltage line. However, there is no indication of a defective solder ball in the x-ray image which was taken at the HLL after the mounting of the passive components (see Figure 8.6).

Both faults on the tested EMCMs were detected by the JTAG system and verified through functional tests. With an analysis of only the error messages of the JTAG software, it is possible to make a statement about the affected functionality of a module. The exact reason for the fault might not be obvious and the decision, if a rework of a module is possible, must be left to the experts. However, a problem in the digital part of the controlling and readout system is always severe for the functionality of the module.

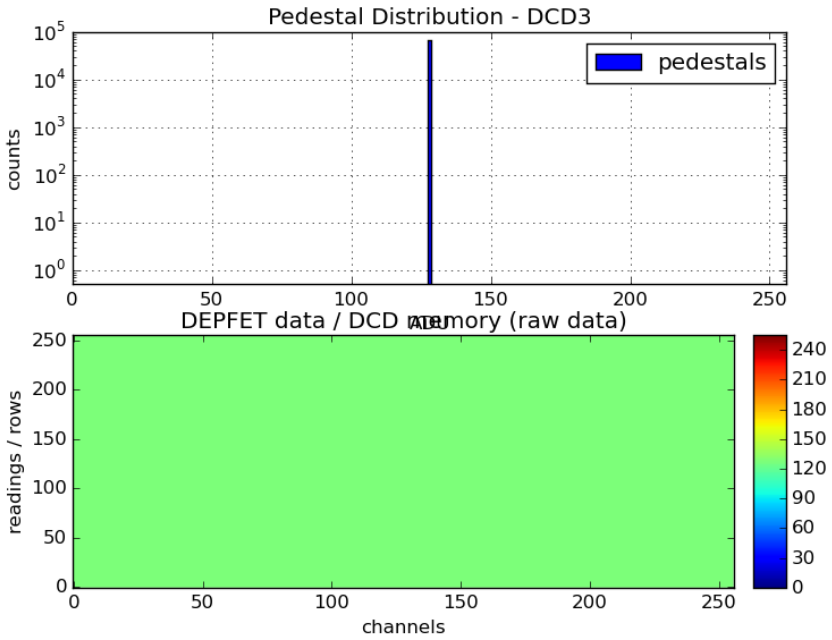


Figure 8.5.: Pedestal readout of the third ASIC pair on EMCM W-17-4. [29]

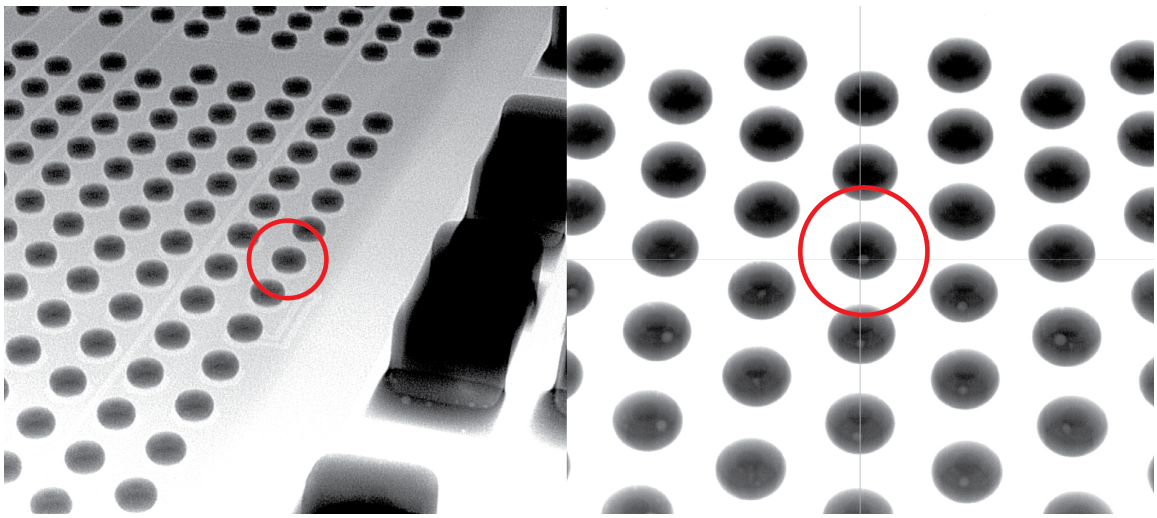


Figure 8.6.: X-ray image of the reference voltage pins of the third ASIC pair on EMCM W17-4. [29]

8.2. PXD9 Modules

The last step before the series production of the final PXD modules is the "pilot run" production of so called PXD9 modules. They are designed as complete functional detector modules with all required dimension. They are populated with all 14 ASICs, which are connected to a big matrix, and the sensitive area is thinned to 75 µm. A picture of a PXD9 modules is shown in Figure 8.7.

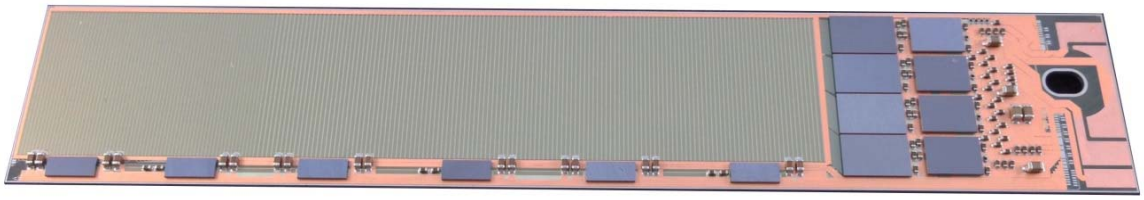


Figure 8.7.: Fully populated PXD9 module with big DEPFET matrix.

One of these modules (PXD9 W30-OB1) was available for a final test of the JTAG system. At the same time, the boundary-scan test served as first test of this new pilot run module. The PXD9 W30-OB1 has a kapton attached, which has also a new design and made a new patch panel design necessary as well. As soon as all components were available, the module was powered for the first time. After the verification that all voltages and currents were in the right range, the ASICs were configured via the JTAG interface using the controlling software of the DHE. The next test was the boundary-scan test. A picture of the setup of the JTAG system is shown in Figure 8.8. The chain integrity check as well as the interconnection test of the JTAG software passed without a single error. It was also possible to create the artificial fault in the Switcher lines as described at the beginning of the chapter. This was a great success for the production of the PXD9 modules as well as for the functionality of the JTAG system.

The PXD9 W30-OB1 is currently further investigated concerning ASIC performance, HS data link stability and matrix functionality. No faults in the digital part of the circuitry occurred so far, confirming the result of the boundary-scan test.

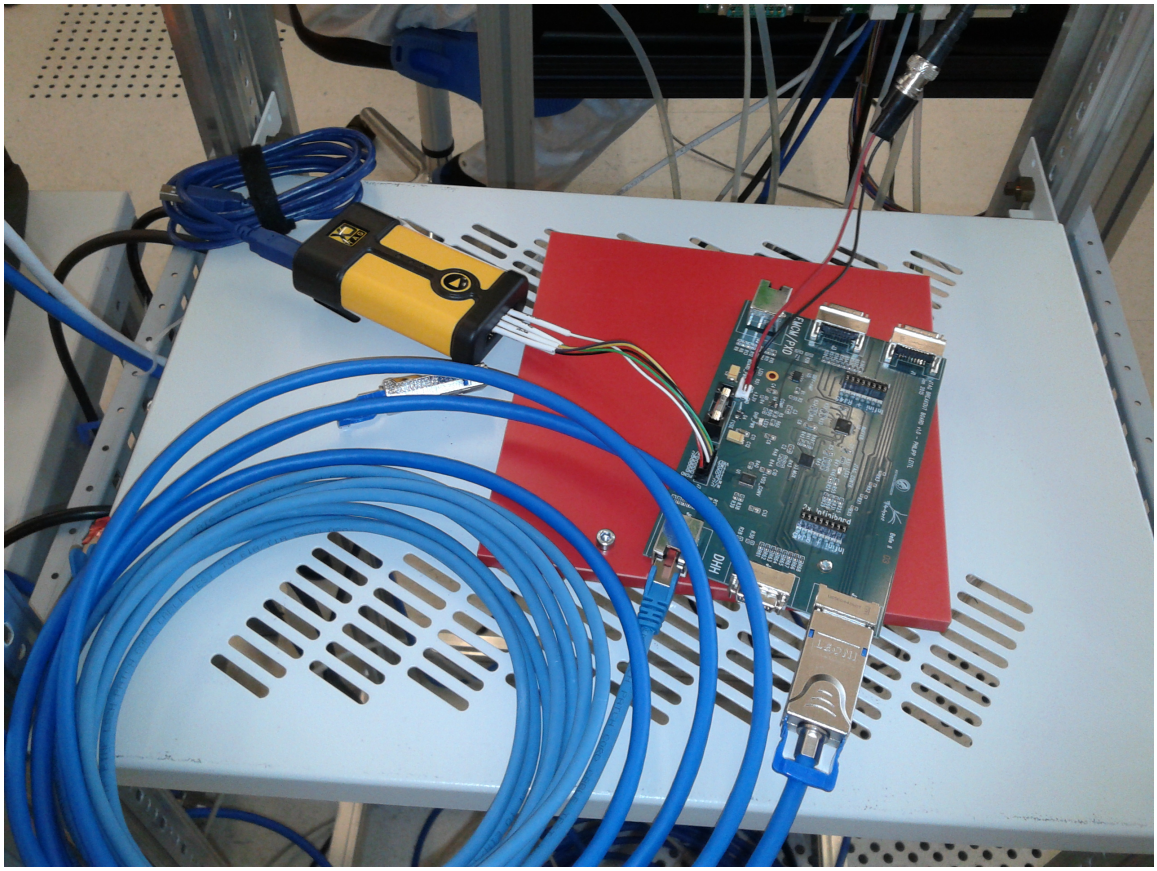


Figure 8.8.: JTAG boundary-scan setup with JBB and XJLink.

9

Conclusion and Outlook: Test Strategy for Series Production

The task of this thesis was the implementation of boundary-scan into the quality assurance of the series production of the PXD modules. The physical preconditions were already given in form of an implementation of the JTAG circuitry into the ASICs, which has already been used successfully for configuration. However, the exact implementations of the boundary-scan circuitry were not sufficiently documented to immediately use a standard JTAG tool. Furthermore, no access to the JTAG interface was provided as it is required for such a tool.

The first step was the development and production of an adapter board to gain access to the JTAG lines. A prototype was built and minor bugs fixed so that eleven boards of the first version of this JTAG Breakout Board could be assembled fulfilling all requirements. The JBB and its functionality were described and documented.

After the study of boundary-scan literature and the according IEEE Std 1149.1, it was possible to develop the required BSDL files for the three ASIC types, considering different versions of them. The access to the actual hardware made it possible to monitor the progress step-by-step and was important for the understanding of the correlations between JTAG software, BSDL files and netlist. It was also necessary to adapt the netlist file, which was available only in the problematic EDIF format. All input files were optimized for the XJTAG software.

The combined JTAG system was successfully tested with the available EMCM and PXD9 modules. It was possible to perform JTAG chain integrity and interconnection tests. The results proofed the ability of boundary-scan to detect faults in the digital communication

of the ASICs. All detected faults could be verified by functional tests. It was also shown that the developed JTAG system can be operated with a needle card setup as it is required for the series production modules.

Recently, a change in the routing of the final PXD modules was done to minimize cross talk effects in the data lines between DCD and DHP. This change has no impact on boundary-scan as it does not touch the logical assignment of the ASIC signals. Because of a change in the bond pad layout, a new needle card must be provided and is already under design to match the new modules. The newest versions of the ASICs were produced by the end of 2015 and were therefore not yet available for boundary-scan tests. The found discrepancies to the standard (ID-Code and inverted clock of the DCD) were considered in the new designs. But it is not expected that these changes will make a reconfiguration of the JTAG system necessary. Therefore, the dedicated software project, which was prepared and already tested with the PXD9 module, can be used in its actual state for the series production.

Although the yield in the module production is very promising, it is necessary to have enough spare modules so that only the modules with the best performance can be chosen for the final detector. In total, 40 modules will be assembled to built the PXD and therefore over 100 modules will be produced in the series production. A fast preselection is required to select the modules which will undergo further functional and performance tests. Boundary-scan is the appropriate tool for this task, as it is able to find all severe problems in the digital communication of the ASICs with a pure test execution time of less than two seconds. The overall testing time for one module is estimated to be shorter than ten minutes and will decrease with experience in probe card handling.

The development of the boundary-scan system and its implementation in the quality assurance was achieved in time. It is ready to be used during the series production beginning in December 2015 and it is prepared to make its contribution to a successful assembly of the PXD for the Belle II experiment.

Acknowledgements

I want to thank everyone who supported me during this work, although I can not name everyone here!

Many thanks got to my supervisors **Hans-Günther Moser** and **Christian Kiesling** for giving me the opportunity to work at this great project. I really learned a lot during the last year!

I want to thank **Markus Fras** and **Miriam Modjesch** for their support concerning hardware design and production issues.

Thanks to all engineers of the electronic production of the MPP, who were involved in the JBB assembly. They did a great job!

Many, many thanks also to my colleges, who supported me during lab tests and could always help with their large experience: **Felix Müller**, **Christian Koffmane**, **Manfred Valentan** ...

A special thank you goes to **Marca Boronat**, who spent hours of strenuous debugging with me in the laboratory. It is always better, if you are not alone!

I also want to thank **Martin Ritter**. He had always time for my questions and easily solved all smaller and bigger PC problems.

I want to thank the hole Belle group for the nice time. It was really a pleasure to work with all of you.

Very special thanks go to my parents, who always supported me on my way. Thank you for giving me the opportunity to fully concentrate on my studies.

The biggest gratitude goes to my girlfriend **Sonja Stindt**, who always is a great help for me and took every burden from me so that I could concentrate on my work. Thank you so much for working through my thesis and thanks for every of the uncountable commas!

Part IV.

Appendix

A

BSDL Files

A.1. DHPT

https://www.mpp.mpg.de/ phleitl/PXD_JTAG_BoundaryScan/BSDL/DHP02.bsdl
https://www.mpp.mpg.de/ phleitl/PXD_JTAG_BoundaryScan/BSDL/DHPT10.bsdl

A.2. DCD

https://www.mpp.mpg.de/ phleitl/PXD_JTAG_BoundaryScan/BSDL/
DCD-B4-Pipeline.bsdl

A.3. Switcher

https://www.mpp.mpg.de/ phleitl/PXD_JTAG_BoundaryScan/BSDL/
switcherb18v2.bsdl

B

Netlist

https://www.mpp.mpg.de/ phleitl/PXD_JTAG_BoundaryScan/netlist/edif_E_Module3_netlist_fullChain_fix.edif

C

XJTAG Project Files

https://www.mpp.mpg.de/ phleitl/PXD_JTAG_BoundaryScan/EMCM.zip
https://www.mpp.mpg.de/ phleitl/PXD_JTAG_BoundaryScan/PXD9.zip

Acronyms

ADC Analog-to-Digital-Converter. 14

ASIC Application-Specific Integrated Circuit. 13–15, 17, 19, 20, 25, 26, 29, 31–34, 37–41, 43, 44, 46–49, 51–53, 59–61, 64, 76–78, 81, 83, 84

BGA Ball Grid Array. 19, 20, 41, 43, 47, 60

BSC Boundary-Scan Register Cell. 34, 36, 38, 39, 47, 61

BSDL Boundary-Scan Description Language. 39–44, 46–48, 50, 53, 56, 59, 75, 78, 83

CCD Charge-Coupled Device. 12

CMOS CMOS: Complementary Metal-Oxide-Semiconductor, a technology for constructing integrated circuits. 17, 52

DCD Drain Current Digitizer. 14–17, 36, 38, 46–48, 52, 53, 59, 61, 76, 77, 79, 84

DEPFET DEpleted P-channel Field Effect Transistor. 9, 11–15, 17, 18, 49, 76, 97

DFT Design for Test. 38

DHC Data Handling Concentrator. 18

DHE Data Handling Engine. 16–18, 63–68, 78, 79, 81

DHH DEPFET Handling Hub. 18

DHP Data Handling Processor. 15–19, 36, 46, 47, 52, 53, 59, 61, 63–66, 68, 75–77, 79, 84

DR Data Register. 29, 31, 36

EDIF Electronic Design Interchange Format. 50, 58, 83

EMCM Electrical Multi-Chip Module. 76–79, 83

GCK Global Reference Clock. 63

HLL Semiconductor Laboratory of the Max Planck Institute. 19, 20, 49, 78, 79

HS High Speed. 65, 67, 81

I/O Input/Output. 34, 36, 41, 61

IC Integrated Circuit. 13, 17, 25, 27, 32, 41, 42, 50, 56, 60, 97

IEEE Institute of Electrical and Electronics Engineers. 26, 27, 31, 39, 47, 48, 59, 83

IR Instruction Register. 29, 31–33, 37, 39, 44, 56

IZM Fraunhofer Institute for Reliability and Microintegration. 19

JBB JTAG Breakout Board. 64–68, 75, 78, 83

JTAG Joint Test Action Group. 16, 18, 19, 26, 27, 29, 31–33, 38, 39, 41, 42, 46, 47, 50, 52, 53, 55–59, 61, 63–66, 68, 75–79, 81, 83, 84

LSB Least Significant Bit. 31, 32, 45, 47

LVDS Low Voltage Differential Signaling. 16, 18, 43, 46, 47, 52, 59, 61, 63, 64, 66, 79

LVTTL Low-Voltage-TTL. 52, 64

MOSFET Metal-Oxide-Semiconductor Field Effect Transistor. 11, 13

PCB Printed Circuit Board. 49, 50, 65, 67

PXD Pixel VerteX Detector. 9, 11–14, 17–19, 25, 26, 32, 34, 36, 40, 48–52, 55, 56, 58–61, 64, 76, 78, 81, 83, 84

SEL Select. 66, 68

SM Standard Model of particle physics. 4, 5

SMD Surface-Mount Device. 19, 20, 67

TAP Test Access Port. 29, 31–34, 36, 37, 39, 41, 43, 44, 47, 57, 63, 64

TCK Test Clock. 29, 32, 37, 38, 43, 47, 57, 63

TDI Test Data In. 29, 31–34, 36, 37, 52, 53, 59, 63

TDO Test Data Out. 29, 31–34, 36, 37, 52, 53, 59, 63, 66

TMS Test Mode Select. 29, 32, 33, 37, 63

TRST Test Reset. 29, 32, 43

VHDL VHSIC Hardware Description Language. 40, 42

Bibliography

[1] ATLAS Collaboration, *Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC*, Phys. Lett. B, 716(1) (2012) 1 – 29, [arXiv:1207.7214]. 4

[2] Wikipedia, *Standard Model — Wikipedia, The Free Encyclopedia* (2015), [Online; accessed 12-October-2015]. 4

[3] Griffiths, D., *Introduction to Elementary Particles*, WILEY-VCH, Weinheim, second, revised edn. (2012). 4, 5

[4] Doležal, Z. and Uno, S. (editors), *Belle II Technical Design Report*, physics.ins-det [arXiv:1011.0352]. 5, 10, 11, 12, 14, 15, 18, 19, 20

[5] Valentan, M., *The Silicon Vertex Detector for b-tagging at Belle II*, Ph.D. thesis, Vienna University of Technology (2013). 9

[6] Vos, M. et al., *DEPFET active pixel detectors*, PoS, VERTEX2009:015. 11

[7] Kemmer, J. and Lutz, G., *New detector concepts*, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, 253(3) (1987) 365 – 377, ISSN 0168-9002. 11

[8] Kemmer, J. et al., *Experimental confirmation of a new semiconductor detector principle*, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, 288(1) (1990) 92 – 98, ISSN 0168-9002, proceedings of the Fifth European Symposium on Semiconductors Detectors. 11

[9] DEPFET Collaboration, *19th International Workshop on DEPFET Detectors and Applications*, Indico page of the workshop, 10 - 13 May 2015. 11

[10] P. Avella et al., *Production quality characterisation techniques of sensors and prototypes for the BELLE II Pixel Detector*, 2015 JINST 10 C01049. 11, 12, 19

[11] Mueller, F., *Some aspects of the Pixel Vertex Detector (PXD) at Belle II*, 2014 JINST 9 C10007. 12, 13, 17

[12] Andricek, L. et al., *Laser tests of the DEPFET gated operation*, 2013 JINST 8 C01051. 12

[13] Knopf, J., *Development, Characterization and Operation of the DCDB, the Front-End Readout Chip for the Pixel Vertex Detector of the Future BELLE-II Experiment*, Ph.D. thesis, Ruperto-Carola University of Heidelberg (2011). 13, 14

[14] P. Avella for the DEPFET collaboration, *DEPFET sensors development for the Pixel Detector (PXD) of Belle II*, 2014 JINST 9 C01057. 18

[15] Levit, D., *DHH Updates*, Presentation at the 7th Belle II VXD Workshop and 18th International Workshop on DEPFET Detectors and Applications January 22nd, 2015. Prague. 18

[16] Levit, D., *Tests of DHE V3.2*, Presentation at the 19th International Workshop on DEPFET Detectors and Applications May 12nd, 2015. Kloster Seeon. 19

[17] Cepheiden, *Underfilled Die* (2010), licensed under common over Wikimedia Commons [Online; accessed 20-November-2015]. 20

[18] Parker, K., *The Boundary-Scan Handbook*, Springer, New York, 3rd edn. (2003). 27, 28, 29, 30, 34, 35, 36, 37, 38, 40

[19] *IEEE Standard Test Access Port and Boundary-Scan Architecture*, IEEE Std 1149.1^{TM}-2001 (R2008). 29, 31

[20] Wenzel, T. and Borowski, M., *Einsamer Boundary Scan IC sucht Anschluss*, Fachartikel, GÖPEL electronic, 2015. 37

[21] *IEEE Standard for Test Access Port and Boundary-Scan Architecture*, IEEE Std 1149.1^{TM}-2013 (Revision of IEEE Std 1149.1-2001). 39, 40

[22] Lemarenko, M. and Germic, L., *DHPT 1.0 Manual*, version 0.1, July 23, 2014. 46, 53

[23] van Dijk, R., *PGA Package* (2005), [Online; accessed 26-November-2015]. 47

[24] Fischer, P., Perić, I. and Kreidl, C., *SwitcherB18 (Gated Mode) Reference Manual*, document revision: 3.2 for chip version 2.0 February 17, 2014. 47

[25] Perić, I. et al., *DCD-Bv2 Reference Manual*, document revision: 0.1 for chip version 2.0 April 9, 2012. 47

[26] Wikipedia, *EDIF — Wikipedia, The Free Encyclopedia* (2015), [Online; accessed 26-November-2015]. 50

[27] Ltd., X., *XJTAG Software — www.xjtag.com* (2015), [Online; accessed 27-November-2015]. 55

[28] Ltd., X., *XJTAG Hardware — www.xjtag.com* (2015), [Online; accessed 27-November-2015]. 56, 58

[29] Boronat, M., *Probe Card Status*, Presentation at the 8th Belle II VXD Workshop, September 10th, 2015. Trieste. 78, 80

Erklärung

Hiermit erkläre ich, die vorliegende Arbeit selbständig verfasst zu haben und keine anderen als die in der Arbeit angegebenen Quellen und Hilfsmittel benutzt zu haben.

München,