# Errata

Unfortunately some errors were missed during proofreading of the PAnIC thesis. This is a summary of the errors and their corrections.

| | | | |
|---|---|---|---|
| Page 4, 1.7, line 3: | module | => | modules |
| Page 13, 2.10, line 4, word 4: | noise | => | noise types |
| Page 13, 2.12, line 2 word 3: | aGlobOut0 | => | aGlobOut0-3 |
| Page 14, Table 10, row 3: | aGobIn0-3 | => | aGlobIn0-3 |
| Page 21, Example, line 6: | [0010 0000/010] | => | [0001 0000/010] |
| Page 23, Table 15, row 25: | OTA | => | DiffBuff |
| Page 27, line 2, last word: | module | => | modules |

Page 27, Table 17: Table is wrong, correct table shown below

*Table 1 TOC content*

| address | content | description |
|---|---|---|
| 00000 000 | 0000 0001 | Sample and hold |
| 00000 001 | 0000 0001 | sub nr |
| 00001 000 | 0000 0010 | Comparator |
| 00001 001 | 0000 0001 | sub nr |
| 00010 000 | 0000 0011 | Differential buffer |
| 00010 001 | 0000 0001 | sub nr |
| 00011 000 | 0000 0100 | Bandgap voltage reference |
| 00011 001 | 0000 0001 | sub nr |
| 00100 000 | 0000 0101 | 8-bit DAC |
| 00100 001 | 0000 0001 | sub nr |
| 00101 000 | 0000 0101 | 8-bit DAC |
| 00101 001 | 0000 0001 | sub nr |

Page 30, IRS_Digital, last sentence: Redundant, it should have been deleted.
Page 30, ORS_Digital, second last sentence: Redundant, it should have been deleted.
Page 31, Table 24, last row: Redundant row, it should have been deleted.
Page 39, S 7.1, line 2: missing comma after “SPI interface”
Page 46, TOC verification, last line: “000100 000” => “00100 000”

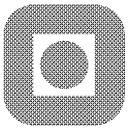
Hovedoppgave ved Institutt for Fysikalsk Elektronikk

# Programmable Analog Integrated Circuit w/ table of content

Carsten Wulff

NTNU 2002

NORGES TEKNISK-NATURVITENSKAPELIGE UNIVERSITET
FAKULTET FOR INFORMASJONSTEKNOLOGI, MATEMATIKK OG ELEKTROTEKNIKK



# HOVEDOPPGAVE

| | |
|---|---|
| Kandidatens navn: | Carsten Wulff |
| Fag: | Fysikalsk elektronikk |
| Oppgavens tittel (norsk): | **Programmable analog integrated circuit (PAnIC) w/ table of content** |
| Oppgavens tittel (engelsk): | **Programmable analog integrated circuit (PAnIC) w/ table of content** |

Oppgavens tekst:

Research has been done on remote laboratories at NTNU since the late 1990's. The Next Generation Laboratory, developed at NTNU during the summer of 2001, has limitations on the number of circuits available for measurement. The assignment is to develop an integrated circuit that can provide flexibility to our remote laboratory through circuit programmability. The project is to be based on a SystemC model developed in the PAIC project. The key criteria of the project are the three features:

- Selection of different circuits to measure
- Ability to combine circuits to form larger circuits
- Self-consistency through a table of content

The operation of the PAnIC is to be verified through simulations and the final layout is to be submitted for prototype production.

| | |
|---|---|
| Oppgaven gitt: | 15. januar 2002 |
| Besvarelsen leveres innen: | 19. november 2002 |
| Besvarelsen levert: | 19.november 2002 |
| Utført ved: | Institutt for fysikalsk elektronikk |
| Veileder: | Trond Ytterdal |

Trondheim, 19. november 2002

Trond Ytterdal

# Abstract

The specification, design, implementation and verification of a programmable analog integrated circuit (PAnIC) with table of content is presented. The architecture of PAnIC is explained and proven through simulations to be a self-consistent programmable analog integrated circuit. The PAnIC offers extended flexibility, through circuit programmability, to our remote laboratory concept [1].

# Preface

This master thesis describes the development of a programmable analog integrated circuit (PAnIC) with table of content from a SystemC [2] model to tape-out. The SystemC model was written as part of a pre-project to PAnIC called PAIC [3]. The purpose of the PAIC project was to develop a viable architecture for a programmable analog integrated circuit with table of content. Carsten Wulff has carried out the PAnIC project at the Norwegian University of Science and Technology, Department of Physical Electronics. The purpose of the project is stated below:

*Develop an integrated circuit based on the PAIC SystemC model. It should contain cells that allow it to "build" an analog to digital converter (ADC)*

There are many people that deserve thanks for their contribution. These are some of them:

- Anita Melvold for always providing essential support when there was no end in sight.
- Trond Ytterdal for always being available for questions, for getting the idea for PAnIC and for funding the PAnIC production
- The students in Analog CMOS2 course for designing the IP modules for PAnIC.
- Tore Barlindhaug for asking the right questions and giving good answers.
- Alf-Egil Edvardsen & Andrew C.O. Wandera for bringing the analog cells to tape-out readiness and sparring on Mentor Graphics.
- Kristin Wulff for proofreading and providing a fresh perspective.
- Egil Wulff for coming up with the name PAnIC.

Carsten Wulff (carsten@wulff.no), November 2002

# Table of content

# List of figures

# List of tables

# CD-ROM content

The PAnIC CD-ROM contains the following:

- PAnIC Thesis (this document)
- PAnIC GDS – II
- PAnIC schematics (as EPS)
- PAnIC VHDL source
- PAnIC SPICE source
- PAIC report
- Mixed - Signal design using Mentor Graphics
- Next Generation Lab - a solution for remote characterization of analog integrated circuits (ICCDCS 2002).
- Programmable analog integrated circuit for use in remotely operated laboratories (ICEE 2002).

# 1. Introduction

This chapter gives an introduction to the PAnIC project and the essentials to understand the architecture. We will start by presenting the motivation and background for the project.

## 1.1 Motivation & Background

Laboratory experience is considered an important part of the curriculum when educating designers of analog (and digital) integrated circuits. Providing laboratory experience to students, especially in the more advanced courses, is costly for a university. This is mostly due to the cost of acquiring measurement equipment that holds an industrial standard. If a university were to equip a laboratory to serve 30 students the cost could easily run into the millions. For example, a Rode & Swartz Vector Network Analyzer for measuring the frequency response of circuits costs around 400 kNOK, purchasing 15 of these, assuming 30 students working in pairs, would come to 6 MNOK. Few universities would consider using that amount of money on a laboratory used by one course. Remotely operated laboratories provide a cost advantage to the conventional laboratory. In a remote laboratory we can use one set of instruments, instead of 15, and still provide students with concurrent access. Remotely operated laboratories have been explored in [1], [4]-[10]. Already there are remote laboratories that enable a student to make measurements on integrated circuits over the Internet. These laboratories often have a limitation on what types of integrated circuits or devices the student has access to.

The Next Generation Lab (NGL) [1,10] was developed at Department for physical electronics at NTNU during the summer of 2001. The NGL can currently measure the frequency response of nine operational amplifiers. The NGL has been presented at the International Caracas Conference on Devices, Circuits and Systems (ICCDCS-2002), the paper [1] is included in Appendix VI. Towards the end of the development there was discussion on were to go next. We wanted to provide students with greater freedom by allowing them to select from a wide range of circuits. It was possible to do this by using switching matrices, as used in LabOnWeb [9], but these

are quite expensive and we wanted to explore other possibilities. In addition we wanted students to be able to create a larger circuit by combining the selectable circuits.

We decided that it should be possible to do this with a programmable analog integrated circuit.

## 1.2 Programmable analog integrated circuits

The concept of a programmable analog circuit is to have an integrated circuit with "standard" cells, which can be wired into an analog circuit i.e. a filter or an amplifier. Figure 1 shows a very simple example of a programmable analog circuit. By controlling a routing network that can connect the analog cells to each other, we can "build" analog circuits.

Programmable analog circuits have been reported since the early nineteen nineties. The earliest reference at IEEE is from 1991 [11]. Several manufactures have made programmable analog circuits, among these are Motorola, IPM Inc, Lattice and Anadigm. Several designs of Field Programmable Analog Arrays (FPAA) have been reported [11]-[14], but these are often aimed at a commercial market as an analog counterpart to Field Programmable Gate Arrays (FPGA) for rapid prototyping of analog circuits. The marked for these FPAA have not gained the same momentum as FPGA, this because of the much greater challenges involved in creating a FPAA. One of the main challenges in creating FPAA is the fact that analog circuits do not have a smallest common denominator. Digital circuits can (in theory) be created from NAND gates regardless of the complexity of the circuit. To circumvent this obstacle one can create expert cells [13, 14], where each analog cell has a set of tunable parameters i.e. a filter with tunable cut-off frequency. These expert cells are designed by analog designers and are guaranteed to operate within specification regardless of how they are connected to other cells. It is a modification of this approach that PAnIC has taken.

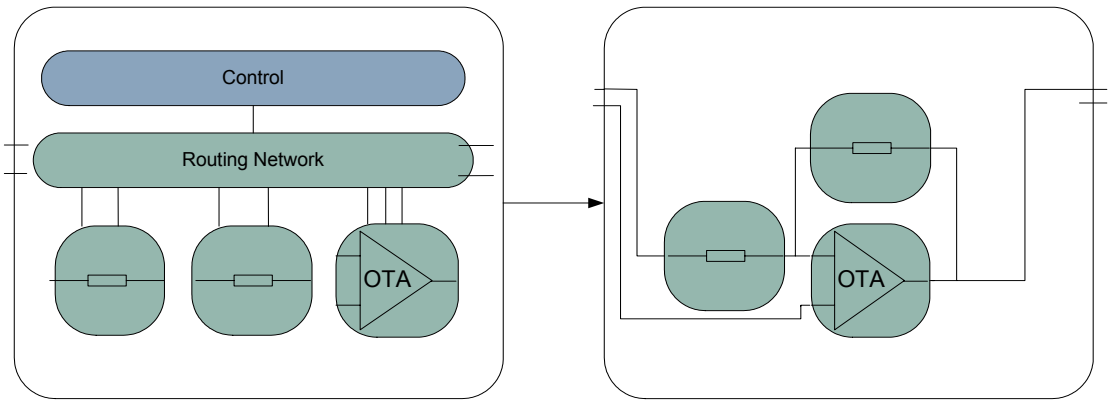



*Figure 1 Programmable analog circuit concept*

## 1.3 System Architecture

The system architecture where the PAnIC chip is going to be used is presented in Figure 2. A web-server is connected to a microcontroller and instruments. The instruments can range from simple multi-meters to advanced network analyzers. The instruments are connected to a circuit board that holds several PAnIC chips. Each PAnIC has a set of analog cells that can be selected alone, or wired together to create a more complex circuit. A student connects to the web-server and gets a graphical user interface that contains a toolbox with all analog cells. The student designs a circuit from the analog cells in the toolbox and submits the circuit to the web-server. The web-server configures the PAnIC chips, through the microcontroller, to create the circuit the student requested. Then it performs measurements on the circuit and returns the result to the student.

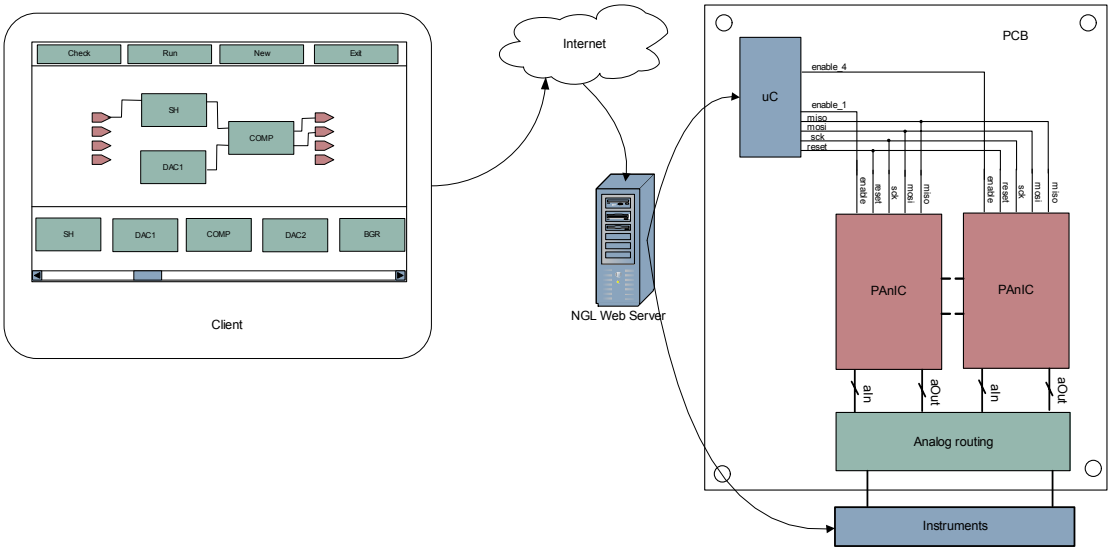



*Figure 2. System Architecture*

## 1.4 Definitions

"Analog cell" is used to describe the analog blocks i.e. a comparator or a sample & hold. An "analog framework" is a framework that consists of several blocks that provide control and input/output switching for the analog cell. "Analog module" is the "analog framework" with an "analog cell".

## 1.5 PAIC project

To investigate possible solutions, a preliminary project [3] was performed during the fall of 2001. The assignment was to develop a viable architecture for a programmable analog integrated circuit that contained information about the analog cells in the circuit and how these cells must be connected to perform a specific function. The concept was to be verified by defining a specific circuit, which was to be modeled and simulated on a functional level. The modeling of the PAIC circuit was done in SystemC and an architecture that met the goals was found. If the reader is unfamiliar with the PAIC architecture it is advisable to read the Design chapter of the project report after reading this introduction. The design chapter has been included in Appendix V for your convenience. The PAIC results were presented at the International Conference on Engineering Education (ICEE-2002), the paper [15] is included in Appendix VII.

## 1.6 PAnIC Project

This project was started January of 2002 and is the continuation of the PAIC project. The project has been renamed to PAnIC, as we felt it was a more suiting name. PAnIC was aimed at taking the SystemC model and creating an integrated circuit, which was to be produced at the end of fall of 2002.

## 1.7 Table of content (TOC)

Each analog module in PAnIC chip has an 8-bit address. This address is used in combination with an 8-bit data packet to control how the module is connected to other module. To know which type of analog cell the module at a specific address contains, the PAnIC has a table of content (TOC). From the system architecture we remember that the PAnIC is to be used by a web-server through a micro-controller. To make this possible the web server has to have information on what analog cells the PAnIC contains. The web-server can ask the micro-controller to retrieve a list of the analog cells and their respective addresses. It can store this information locally on disc or other storage for later use. Each entry is an 8-bit number, which defines the analog cell type, and an 8-bit sub number that defines a specialization of the analog cell type. The web server must thus contain a look-up table where a complete definition of the analog cell resides.

## 1.8 Learning PAnIC connections

In addition to knowing the type of analog cell and which address it resides at, the web-server needs to know what connections the PAnIC chip can perform. The PAnIC provides a feature, called “ReadBack”, that allows the micro-controller to read the connections between the analog modules. Normally the analog cell will be connected through an output register & switch (ORS) to an input register & switch (IRS) of another analog module in the PAnIC. A simple algorithm in the micro-controller can read this connection.

*Pseudo code for reading back physical connections:*

```
reset PAnIC
foreach module{
   foreach ORS{
       Set output low
   }
}

foreach module{
   foreach IRS{
       foreach module{
          foreach ORS{
              Set output high
              Read from IRS
              if(data from IRS contains a one){
                store output
              }
              Set output low
          }
       }
   }
}
```

This algorithm is time consuming since it has to iterate through each IRS and ORS on the chip, but with minimal programming in the micro-controller it can provide the web server with information on all connections inside the PAnIC. After reading the list the web server can store the information on disk for faster access.

# 2. Specification

This chapter presents the specification of the PAnIC chip.

## 2.1 Pin description

This section describes the pins and their function. Figure 3 shows the PAnIC symbol and Table 1 contains the pin description. The power pins have only been shown once, but there shall be 3 VDD pins, 3 VDD3V pins and 3 VSS pins to provide enough current and stable supply voltage to PAnIC.

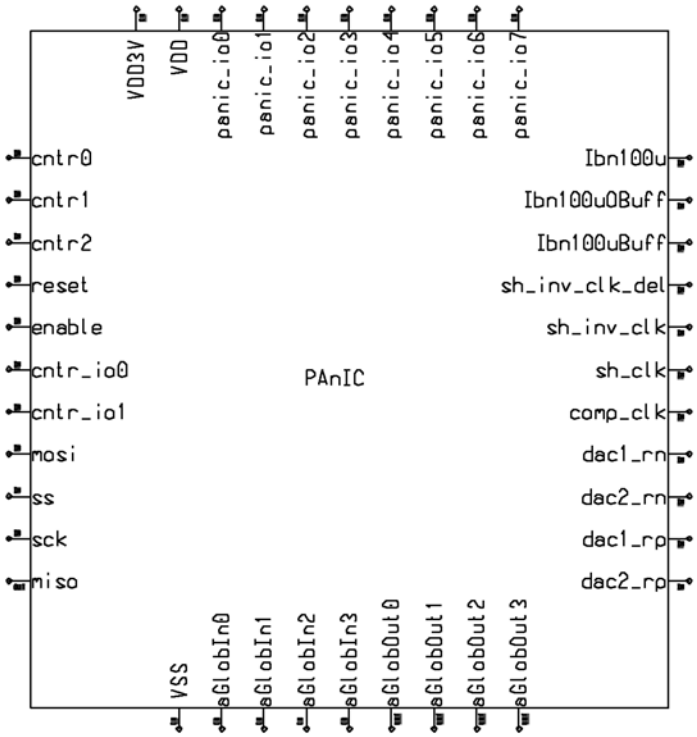



*Figure 3 PAnIC symbol*

*Table 1 Pin description*

| Pins | Direction | Type | Function |
|---|---|---|---|
| cntr0 | in | digital | Control signal 0 |
| cntr1 | in | digital | Control signal 1 |
| cntr2 | in | digital | Control signal 2 |
| reset | in | digital | Active low reset |
| enable | in | digital | Active high enable |
| cntr_io0 | in | digital | Secondary data interface control signal 0 |
| cntr_io1 | in | digital | Secondary data interface control signal 1 |
| mosi | in | digital | Primary data interface, master out, slave in |
| ss | in | digital | Primary data interface, start transmission |
| sck | in | digital | Primary data interface, transmission clock |
| miso | out | digital | Primary data interface, master in, slave out |
| VSS | | power | Ground |
| aGlobIn0 | in | analog | Analog input 0 |
| aGlobIn1 | in | analog | Analog input 1 |
| aGlobIn2 | in | analog | Analog input 2 |
| aGlobIn3 | in | analog | Analog input 3 |
| aGlobOut0 | out | analog | Analog output 0 |
| aGlobOut1 | out | analog | Analog output 1 |
| aGlobOut2 | out | analog | Analog output 2 |
| aGlobOut3 | out | analog | Analog output 3 |
| dac2_rp | in | analog | DAC 2 positive reference voltage |
| dac1_rp | in | analog | DAC 1 positive reference voltage |
| dac2_rn | in | analog | DAC 2 negative reference voltage |
| dac1_rn | in | analog | DAC 1 negative reference voltage |
| comp_clk | in | digital | Comparator clock input |
| sh_clk | in | digital | Sample & hold clock input |
| sh_inv_clk | in | digital | Sample & hold inverted clock input |
| sh_inv_clk_del | in | digital | Sample & hold delayed inverted clock input |
| Ibn100uBuff | in | analog | Internal buffer bias current input, 100µA |
| Ibn100uOBuff | in | analog | Output buffer bias current input, 100µA |
| Ibn100u | in | analog | Analog cell bias current input, 100µA |
| panic_io7 | inout | digital | Secondary data interface signal 7 (MSB) |
| panic_io6 | inout | digital | Secondary data interface signal 6 |
| panic_io5 | inout | digital | Secondary data interface signal 5 |
| panic_io4 | inout | digital | Secondary data interface signal 4 |
| panic_io3 | inout | digital | Secondary data interface signal 3 |
| panic_io2 | inout | digital | Secondary data interface signal 2 |
| panic_io1 | inout | digital | Secondary data interface signal 1 |
| panic_io0 | inout | digital | Secondary data interface signal 0 (LSB) |
| VDD | | power | PAnIC power |
| VDD3V | | power | Analog cell power |

## 2.2 Data Interface

The data interface is an essential part of the PAnIC architecture. Without a functioning data interface the PAnIC is useless. Therefore, the PAnIC shall be designed with two separate and independent data interfaces. The primary data interface shall be a Serial Peripheral Interface (`SPI`), which is a commonly used interface in micro-controllers (explanation of the interface in Appendix II). The second data interface shall be an 8-bit input/output bus. The direction of this bus and what signal it taps into shall be controlled by `cntr_io0` and `cntr_io1`.

## 2.3 Control

The state of PAnIC shall be controlled through a 3-bit control bus (`cntr`). The state shall decide what function PAnIC performs. Table 2 describes the different signals and their corresponding state. Two of the states have an 8-bit data packet associated with them, these are “load address” and “load data” the other two produce data that can be read through the data interface.

*Table 2 Control signals*

| **cntr** | **State** | **Description** |
|---|---|---|
| 000 | IDLE | No operation |
| 001 | Load address | Loads the current data value into the address register |
| 010 | Load data | Loads the current data into the register defined by the current address |
| 011 | ReadBack | Does a readback of the IRS defined by the current address |
| 100 | TOC | Reads the TOC word at the location defined by the current address |
| 101 | IDLE | No operation |
| 110 | IDLE | No operation |
| 111 | IDLE | No operation |

## 2.4 Routing Network

The routing network connects the analog cells to the `aGlobIn0-3`, `aGlobOut0-3` signals and to each other. The routing network consists of input switches, output switches, internal buffers and output buffers. The routing network specifications shown in Table 3.

*Table 3 Routing network specification*

| **Name** | **Spec** |
|---|---|
| Signal swing | 0-3V |
| Internal buffer capacitive load | < 5pF |
| Output buffer capacitive load | < 50pF |
| Internal buffer resistive load | No resistive load |
| Output buffer resistive load | No resistive load |
| Internal buffer slewrate | > 8V/µs |
| Output buffer slewrate | > 8V/µs |
| Routing network bandwidth | > 30MHz |
| Phase shift up to 1MHz | < 5 degrees |
| Phase shift at 30MHz | < 95 degrees |
| Switch damping when off | > 50dB |

## 2.5 Analog cells

Students in the course Analog CMOS 2 during the spring of 2002 designed the analog cells that were going to be used in PAnIC. These were a Differential Buffer, a Sample & Hold, a Bandgap voltage Reference, a Comparator and an 8-bit DAC. The DAC designed by the students was not finished when it needed to be because of problems with the computer aided design software. There was no time to complete them after the software problems were fixed, instead we chose to use an 8-bit DAC included in the analog cell library of the foundry. A description of each analog cell follows.

### Differential Buffer (DiffBuff)

The differential buffer is an operational transconductance amplifier with four capacitors. The buffer has a –6dB dampening up to 10MHz and the –9dB frequency is at 80MHz. A simplified schematic is shown in Figure 4. The port map for the differential buffer is shown in Table 4.

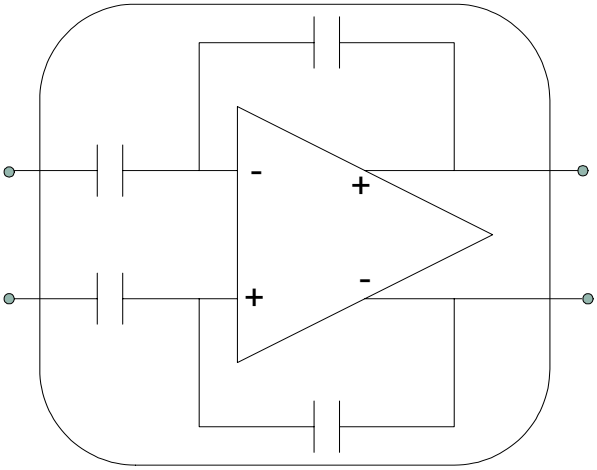



*Figure 4 Differential buffer diagram*

*Table 4 Differential buffer ports*

| Port | Direction | Type | Description |
|---|---|---|---|
| vin_p | in | analog | positive input |
| vin_n | in | analog | negative input |
| ibp5u | in | analog | 5uA bias current |
| vout_p | out | analog | positive output |
| vout_n | out | analog | negative output |
| vdda | | power | 2.7 – 3.3V |
| vssa | | power | ground |

## Sample & Hold (SH)

This circuit is implemented with clocked folded cascode operational amplifiers. It is pseudo differential and can be used as a single ended or differential sample & hold. The port map is shown in the table below.

*Table 5 Sample & Hold ports*

| Port | Direction | Type | Description |
|---|---|---|---|
| vdda | | power | 2.7 – 3.3 V |
| vssa | | power | Ground |
| vin_n | in | analog | Negative input |
| vin_p | in | analog | Positive input |
| Ibp5u | in | analog | 5uA bias current |
| inv_clk_del | in | digital | Delayed inverted clock |
| inv_clk | in | digital | Inverted clock |
| clk | in | digital | Clock |
| vout_n | out | analog | negative output |
| vout_p | out | analog | positive output |

## Bandgap Voltage Reference (BGR)

The bandgap voltage reference is based on two diode connected NMOS transistors and delivers a reference voltage of 1.2V +- 40mV within the range –40 to 85 degrees Celsius. It also delivers a reference voltage that is proportional to absolute temperature (0.6V at room temperature). The table below describes the ports of the bandgap voltage reference.

*Table 6 Bandgap voltage reference ports*

| Port | Direction | Type | Description |
|---|---|---|---|
| vdda | | power | 2.7 – 3.3 V |
| vssa | | power | ground |
| vref | out | analog | reference voltage |
| v_ptat | out | analog | reference proportional to absolute temperature |

## Comparator (COMP)

The comparator is a latched comparator with differential output. The circuit consists of a fully differential amplifier followed by a gain stage and by a positive feedback latch in a track and latch configuration. The resolution is 0.5 mV in a range between 765 mV and 2.77V with a power supply of 3V. The table below describes the ports of the comparator.

*Table 7 Comparator ports*

| Port | Direction | Type | Description |
|---|---|---|---|
| vdda | | power | 2.7 – 3.3 V |
| vssa | | power | ground |
| vin_p | in | analog | positive input |
| vin_n | in | analog | negative input |
| ibn5u | in | analog | 5uA bias current |
| clk | in | digital | Sampling clock |
| vout_n | out | analog | negative output |
| vout_p | out | analog | positive output |

### 8-bit digital to analog converter (DAC)

The DAC is based on two resistor dividers. The signals `rn` and `rp` set the negative and positive range of the DAC. The output is given by `Vdacout` = (`Vrp` – `Vrn`)/`256 * code in` + `Vrn`. The table below describes the ports of the DAC.

*Table 8 DAC ports*

| Ports | Direction | Type | Description |
|---|---|---|---|
| dacin(7:0) | in | digital | digital input |
| dacout | out | analog | analog output |
| rn | in | analog | negative reference |
| rp | in | analog | positive reference |
| vdda | | power | 4.5 – 5.5 V |
| vssa | | power | ground |

## 2.6 Process

The PAnIC shall be produced in AMS 0.6µm mixed mode twin-well CMOS process with three layers of metal, high resistive poly and double poly capacitors. The process is made available through a Multi-Project Wafer (MPW) run organized by Europractice [16].

## 2.7 Area

The area of the finished PAnIC is not critical, but it should not exceed 10 $mm^2$, which is the minimum area of a design in the MPW run.

## 2.8 Power

The power consumption of PAnIC will not be specified, but calculations are to be performed to ensure that the PAnIC layout can supply the cells with the power they need. Two single ended power supplies shall power PAnIC. The first is for the analog cells and the second is for powering the rest of PAnIC. The table below describes the acceptable power range.

*Table 9 PAnIC acceptable power range*

| Name | Range |
|---|---|
| VDD | 4.8-5.2V |
| VDD3V | 2.8-3.2V |

## 2.9 Speed

No high performance demands are set on the PAnIC, but it should be capable of supporting clock speeds on the COMP module up to 1MHz.

## 2.10 Noise

Noise like thermal noise, charge injection, 1/f, kT/C and shot-noise are not critical for the PAnIC architecture. It is not required that special considerations are made to minimize these noise. Making sure the power supply to each cell remains stable is considered important. Minimizing noise through the substrate is to be taken into consideration, but only to a certain point due to the asynchronous nature of the PAnIC architecture.

## 2.11 Bias currents

Three bias currents shall be provided for PAnIC, all at 100µA. These bias currents supply the analog cells, internal buffers and output buffers separately.

## 2.12 Functional Description

The PAnIC should be able to connect the inputs and outputs of the analog cells to the `aGlobIn0-3` and `aGlobOut0` signals. It should also be capable of connecting some of the analog cells together to form larger circuits. The most significant of these circuits are an 8-bit successive approximation analog to digital converter; an explanation of this architecture is included in Appendix I, which is made from the sample & hold, comparator and a DAC. The analog cells with inputs shall be connected in the manner described in Figure 5. The boxes in front of each analog cell are switches that can connect the signals on the box input to the analog cell. All outputs from analog cells shall be possible to connect to the `aGlobOut0-3` signals.

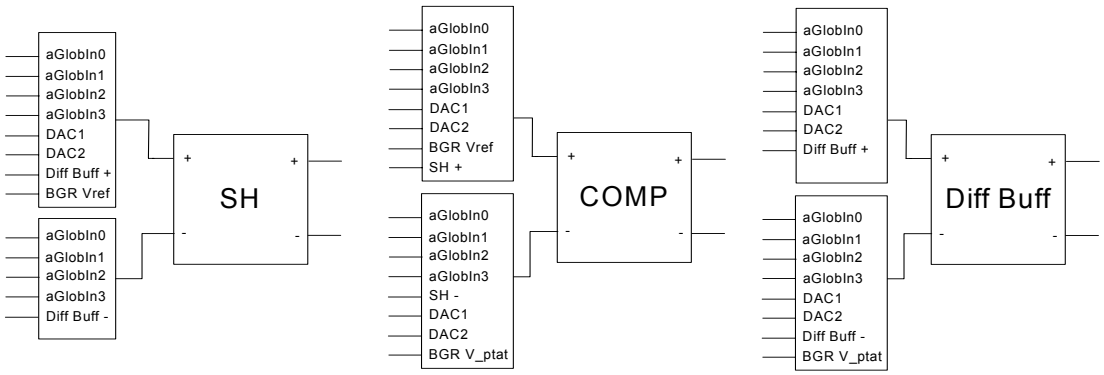



*Figure 5 Possible input connections*

## 2.13 Test description

Table 10 outlines the different tests that are to be performed on the produced PAnIC chip.

*Table 10 Tests to be performed on PAnIC chip*

| **Test** | **Type** | **Description** |
|---|---|---|
| power up | short test | slowly power up power pins and biases to check for shorts |
| Primary data interface | functional | writing and reading to PAnIC through SPI |
| Secondary data interface | functional | writing and reading to PAnIC through `panic_io` bus |
| TOC readout | functional | Reading TOC and verifying content |
| ReadBack | functional /short test | Performing ReadBack to validate ReadBack function and to check connections between analog modules and connections to `aGobIn0-3`. |
| Output test | functional | Using DACs to check output swing and function |
| Analog cell test | functional | Check the function of each analog cell |
| ADC test | functional | Check the ADC without using the microcontroller as a SAR register (Appendix I) |
| ADC full test | functional | Check the operation of the ADC with SAR register |
| Speed up 1 | speed test | Increase ADC speeds to check maximum speed of ADC and routing network. |
| Speed up 2 | speed test | Use a DAC to check max write speed (digital functions) and output slewrate. |
| Frequency response | functional | Use the DiffBuff to test the frequency response of the routing network (aGlobIn -> DiffBuff -> aGlobOut) |
| Current up | stability | Use the DiffBuff in the same setup as frequency response test and check the stability of buffers by adjusting bias currents |
| Check permutations | functional | Check the function of all the different circuits that can be connected using PAnIC. |

# 3. Verification Plan

This chapter details how we are going to verify that the PAnIC conforms to the specification.

## 3.1 Verification of digital functions

The digital cells, written in VHDL, shall be verified on a functional level. Functional test on data interfaces, TOC, ReadBack feature and routing control shall be performed. The synthesized standard cell netlist shall be verified using the same testbench as the VHDL model.

## 3.2 Verification of routing network

The routing network consists of the cells `irs_cellv2`, `ORS_analog` and `outputbuff`, these cells will be explained later. They shall be verified by performing simulations using SPICE. The best, typical and worst value for the parameters in the specification shall be extracted using the typical, worst speed and worst power model parameters of the process.

## 3.3 Mixed-Signal verification

The mixed-signal (analog cells and digital cells combined) cells shall be verified on a functional level using SPICE. Top-level verification of the operation of PAnIC shall be performed on extracted netlists, which includes pads, from both schematics and layout. The top-level tests that shall be performed are; functional verification of all analog cells, functional verification of the ADC, functional verification of the TOC, functional verification of ReadBack feature on analog modules with inputs and functional verification of data interfaces.

## 3.4 Timing Verification

Verification of the timing demands shall be performed on the comparator by checking the time from start of comparation to a valid output signal.

## 3.5 Layout verification

The PAnIC layout shall be verified against the schematic by performing a Layout VS Schematic (LVS) check. A Design Rule Check (DRC) shall be performed to verify that the layout conforms to the design rules of the specific process.

# 4. Design

In this chapter we will give an overview of the design methodology, explain the PAnIC architecture and how to use PAnIC.

## 4.1 Design methodology

The tools and how they are used will not be explained as a part of this thesis. A manual on the use of tools in mixed-signal designs [17] has been written as a direct consequence of this project.

Figure 6 shows the design flow. The project started with a SystemC model that was created as a pre-project to PAnIC. The SystemC model was the basis for the digital cells written in VHDL. The SystemC architecture was modified to create a VHDL code that could be synthesized. After confirming that the digital portions worked, the VHDL code was synthesized with Synopsys. A VHDL standard-cell netlist was extracted from the synthesized design and simulated in ModelSim to ensure that the synthesized design corresponded to the VHDL code.

The standard cell netlist was imported into Design Architect, as schematics, which were used as logic source for automatic place & route. The netlist were converted to schematics because the IC Station version we used could not place & route directly from a standard cell netlist. A second reason was that no mixed-signal simulators (co-simulation of VHDL and SPICE) were available.

The analog portions were designed using SPICE netlists and Eldo. The analog circuits were manually translated into Design Architect schematics. The digital and analog portions were combined using Design Architect. SPICE netlists were extracted from schematics and used to verify the design.

The layout was performed in IC Station. IC Stations place & route was used for the digital portions. Schematic Driven Layout (SDL) was used for analog portions. The layout was verified using Design Rule Check (DRC) and Layout Versus Schematic (LVS). The final design was submitted to the foundry as a GDS-II file.

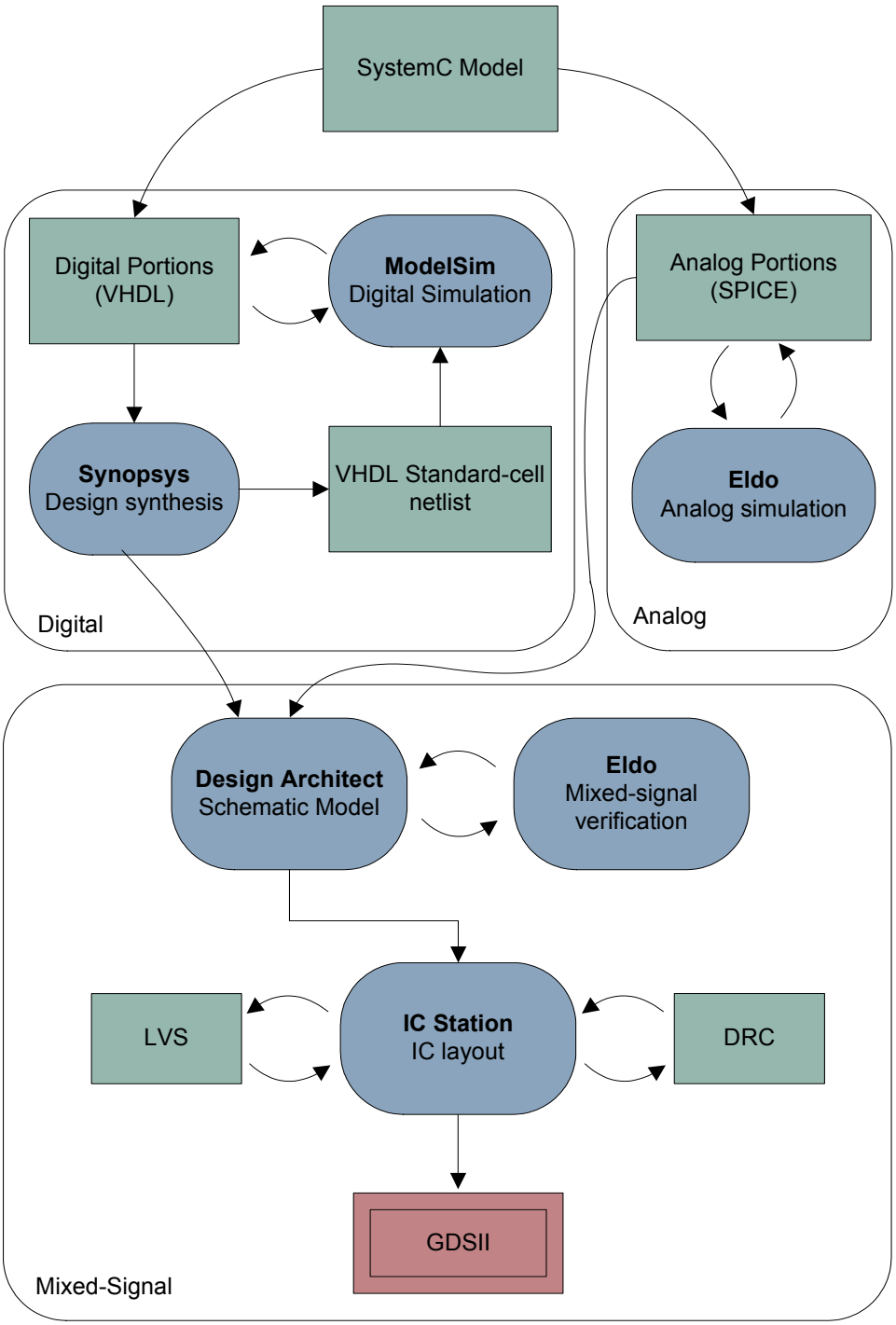



*Figure 6 Design Flow*

## 4.2 PAnIC architecture

The PAnIC contains 6 analog cells; two digital to analog converters, one sample & hold, one comparator, one differential buffer and one bandgap voltage reference. Figure 7 shows a block diagram. The logic that enables programming of the PAnIC is divided into `Panic_Control` and analog module frameworks (AMF). The AMF is the core of routing capabilities in PAnIC. The AMF comes in two types; one for cells with analog inputs and one for cells without. These are called `AMF_LMIR2OR2` (with inputs) and `AMF_LMOR2` (without inputs).

At the input of the `AMF_LMIR2OR2` we have an IRS. This module has 8 analog inputs and 1 analog output. The inputs are connected to switches that are controlled by an 8-bit register, the output from the switches are combined, thus the IRS can switch 8 input signals onto 1 output signal in any order. In the PAIC architecture each AMF had 5 IRS blocks, which meant up to 40 switch able inputs, but during the design of PAnIC the number was reduced to 2 IRS because no more was needed with the analog cells we were going to use. The global inputs `aGlobIn0-3` are connected to the first four inputs of the IRS.

At the output of both AMF we have two ORS, these can connect the output signal from an analog cell to one of the four off-chip signals (`aGlobOut0-3`). A digital input is provided in both AMF by a module register (ModReg).

As mentioned before, each analog module is addressed by an 8-bit word. Bits 0 – 2 we call “line address”, bits 3 – 6 we call “module address” and bit 7 is unused. A 4 to 16 decoder in `Panic_Control` decodes the module address into enable signals for the analog modules. A 3 to 8 decoder in the AMF decode the line address into enable signals for IRS, ORS and ModReg.

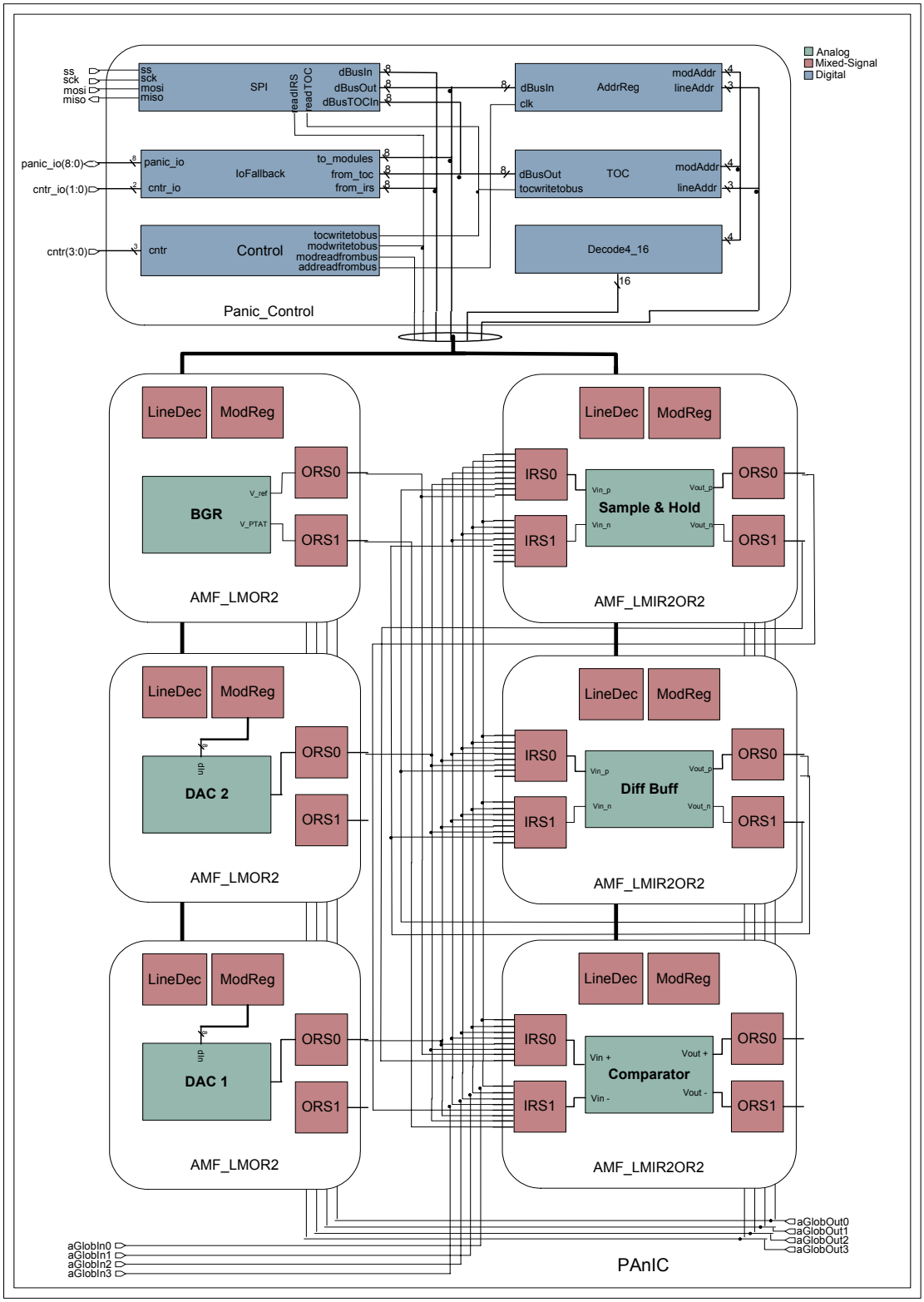



*Figure 7 PAnIC block diagram*

## 4.3 Using PAnIC

This section gives an example of programming PAnIC to further explain the architecture of PAnIC.

The PAnIC is programmed through the SPI data interface and the `cntr` bus (Figure 7). Each programming step consists of a "data packet" and "control signal" pair, the data packet is 8-bit and the control signal is 3-bit. From here on they will be written as [data-packet/control-signal], for example [0000 0001/010]. The control signals are listed in Table 2 (section Control in Specification chapter). To program a connection within the PAnIC we need two pairs ([address/001] and [data/010]). The address is in two parts, one module address and one line address. The module addresses and line addresses are listed in Table 11 and Table 12 respectively, for example writing "0000 0001" to the ORS0 of `DAC1` requires that we first write [00100 000/001] (the address) and then [0000 0001/010] (the data). An address pair will be written on the form [moduleaddress lineaddress/010] to make it easier to recognize. The 8-bit data performs a different function depending on which address it is written to. There can be three "endpoints" for each analog module, an IRS, an ORS or the ModReg. Table 13 lists all endpoints that perform a function when written to, for example writing [00100 010/001] (load address) then [0000 0000/010] (load data) sets `DAC1` to its minimum value and writing [00100 010/001] then [1111 1111/010] sets `DAC1` to its maximum. The ORS functions are common for all analog modules; these are listed in Table 14. The IRS has some common functions and some different, these are listed in Table 15.

We now have all that is needed to control the PAnIC. As an example we will connect the circuit shown in Figure 8, here `DAC2` is connected to the `SH`, the `SH` is connected to the `COMP`, `DAC1` is connected to the `COMP` and the `COMP` output we will connect to the global output. To do this we write:

| | | |
|---|---|---|
| [00000 011/001] | Address for IRS0 of `SH` | (Table 11 & Table 12) |
| [0010 0000/010] | Connect `DAC2` to `SH` | (Table 15) |
| [00001 011/001] | Address for IRS0 of `COMP` | (Table 11 & Table 12) |
| [0001 0000/010] | Connect `SH` to `COMP` | (Table 15) |
| [00001 100/001] | Address for IRS1 of `COMP` | (Table 11 & Table 12) |
| [0010 0000/010] | Connect `DAC1` to `COMP` | (Table 15) |
| [00001 000/001] | Address for ORS0 of `COMP` | (Table 11 & Table 12) |
| [0000 0001/010] | Connect ORS0 output to `aGlobOut0` | (Table 14) |
| [00001 001/001] | Address for ORS1 of `COMP` | (Table 11 & Table 12) |
| [0000 0010/010] | Connect ORS1 output to `aGlobOut1` | (Table 14) |

To check the connection we test if the comparator switches at the right value. We first write a digital word to DAC2, then we step DAC1 from below the digital word in DAC2 to above the digital word in DAC2.

[00101 010/001] Address for ModReg of DAC2
[0000 1111/010] Load the digital word
[00100 010/001] Address for ModReg of DAC1
[0000 1110/010] Load value below, aGlobOut0 is low (from comparator)
[0001 0000/010] Load value above, aGlobOut1 is high (from comparator)

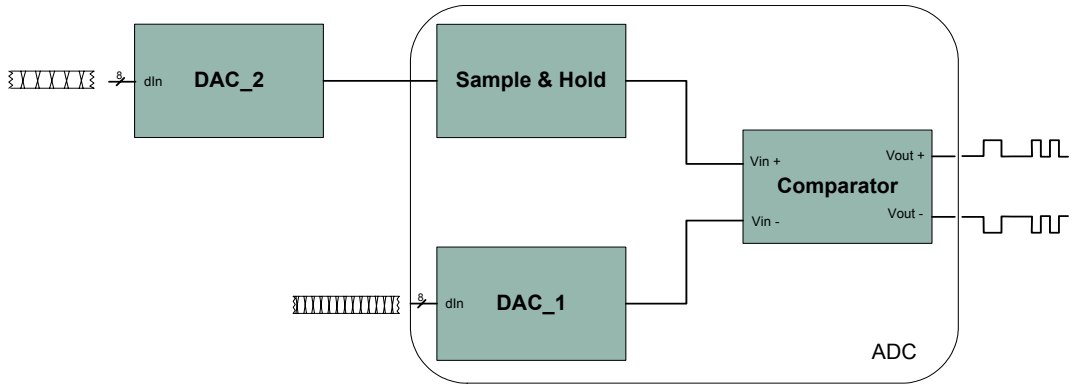



Figure 8 ADC block diagram

*Table 11 Module Address*

| **datapacket(7:3)** | **Module** |
|---|---|
| 00000 | Sample & Hold |
| 00001 | Comparator |
| 00010 | Differential buffer |
| 00011 | Bandgap voltage reference |
| 00100 | DAC 1 |
| 00101 | DAC 2 |

*Table 12 Line Address*

| **datapacket(2:0)** | **Cell** |
|---|---|
| 000 | ORS 0 |
| 001 | ORS 1 |
| 010 | IRS 0 |
| 011 | IRS 1 |

*Table 13 Module endpoints*

| **Cell** | **Endpoints** |
|---|---|
| Sample & Hold | IRS0, IRS1, ORS0, ORS1 |
| Comparator | IRS0, IRS1, ORS0, ORS1 |
| DiffBuff | IRS0, IRS1, ORS0, ORS1 |

| | |
|---|---|
| Bandgap | ORS0, ORS1 |
| DAC 1 | ORS0, ModReg |
| DAC 2 | ORS0, ModReg |

*Table 14 ORS routing functions*

| **datapacket(2:0)** | **ORS input routed to** |
|---|---|
| 000 | ORS output |
| 001 | ORS output & aGlobOut0 |
| 010 | ORS output & aGlobOut1 |
| 011 | ORS output & aGlobOut2 |
| 100 | ORS output & aGlobOut3 |

*Table 15 IRS routing functions*

| **lineaddress** | **datapacket(8:0)** | **IRS output connected to** |
|---|---|---|
| common: | | |
| 011 | 0000 0001 | aGlobIn0 |
| 011 | 0000 0010 | aGlobIn1 |
| 011 | 0000 0100 | aGlobIn2 |
| 011 | 0000 1000 | aGlobIn3 |
| 100 | 0000 0001 | aGlobIn0 |
| 100 | 0000 0010 | aGlobIn1 |
| 100 | 0000 0100 | aGlobIn2 |
| 100 | 0000 1000 | aGlobIn3 |
| Sample & hold: | | |
| 011 | 0001 0000 | DAC 1 |
| 011 | 0010 0000 | DAC 2 |
| 011 | 0100 0000 | DiffBuff positive output |
| 011 | 1000 0000 | BGR_Vref |
| 100 | 0001 0000 | DiffBuff negative output |
| Comparator: | | |
| 011 | 0001 0000 | DAC 1 |
| 011 | 0010 0000 | DAC 2 |
| 011 | 0100 0000 | BGR_Vref |
| 011 | 1000 0000 | Sample & Hold negative output |
| 100 | 0001 0000 | Sample & hold positive output |
| 100 | 0010 0000 | DAC 1 |
| 100 | 0100 0000 | DAC 2 |
| 100 | 1000 0000 | BGR_Vptat |
| OTA: | | |
| 011 | 0001 0000 | DAC 1 |
| 011 | 0010 0000 | DAC 2 |
| 011 | 0100 0000 | Diff buff positive output |
| 100 | 0001 0000 | DAC 1 |
| 100 | 0010 0000 | DAC 2 |
| 100 | 0100 0000 | Diff buff negative output |

# 5. Implementation

The PAnIC chip contains several digital and analog cells. The digital cells were written in VHDL and the analog cells in SPICE.

The VHDL source code for PAnIC can be found on the CD-ROM. The source is divided into two parts, one that contains the code that was used to synthesize the digital portions of PAnIC and the other that was created to test the code.

The SPICE source code for each cell is included as part of the top-level netlist, the netlist can be found on the CD-ROM.

Listed in Table 16 are the cells used in PAnIC. All schematics, except for the analog cells, are included in the Appendix III and are shown in the same order as in Table 16. Each cell in this chapter has a reference to the figure where its schematic can be found. The cells in PAnIC will be explained in following order; Top level, PAnIC Control, AMF and support circuitry.

*Table 16 Cells in PAnIC*

| **Cell name** | **Parent cell** | **Description** |
|---|---|---|
| maspan | | panic_top cell with pads |
| IOA5P | maspan | Analog io pad without series resistance |
| IOA2P | maspan | Analog io pad with series resistance |
| OB33 | maspan | Digital output pad |
| IB15 | maspan | Digital input pad |
| IOF3 | maspan | Digital io pad |
| PP01 | maspan | Power pad (ground) |
| PP02 | maspan | Power pad (VDD) |
| panic_top | maspan | panic cell with output buffers |
| panic_io_buff | panic_top | panic_io(7:0) to panic_in(7:0) and panic_out(7:0) converter |
| outputbuff | panic_top | Analog buffer for driving pads |
| cb_100u | panic_top | Current copier, 1x100µA > 6x100µA |
| panic | panic_top | panic cell |
| panic_io_tribuff | panic_io_buff | 1 bit io to input and output |
| panic_control | panic | panic control |
| AMF_LMIR2OR2 | panic | AMF with input |

| | | |
|---|---|---|
| AMF_LMOR2 | panic | AMF without input |
| sample & hold | panic | Sample & hold |
| comparator | panic | Comparator |
| DAC8 | panic | 8-bit DAC |
| BGR_final | panic | Bandgap Voltage Reference |
| DiffBuff | panic | Differential buffer |
| curr_combine | panic | Current divider 1x10µA > 2x5µA |
| curr_combine_cells | panic | Current divider 1x100µA > 4x5µA |
| curr_combine10u | panic | Current divider 1x100µA > 6x10µA |
| addreg | panic_control | Address register |
| toc | panic_control | Table of content |
| iofallback | panic_control | Backup input/output |
| decode4_16 | panic_control | 4 to 16 decoder |
| spi | panic_control | Serial Peripheral Interface |
| control | panic_control | Control signal decoder |
| AMF_Digital_LMIR2OR2 | AMF_ LMIR2OR2 | Digital portion of AMF with input |
| ORS_analog | both AMF | Analog portion of ORS |
| IRS_analogv2 | AMF_ LMIR2OR2 | Analog portion of IRS |
| AMF_Digital_LMOR2 | AMF_LMOR2 | Digital portion of AMF with output |
| LineDec | both AMF_Digital | 3 to 8 decoder |
| IRS_Digital | AMF_Digital_ LMIR2OR2 | Digital portion of IRS |
| ORS_Digital | both AMF_Digital | Digital portion of ORS |
| ModReg | both AMF_Digital | 8-bit Register |
| irs_cellv2 | IRS_analogv2 | Switch and readback in IRS |
| buff_ors | ORS_analog | Buffer used to drive the routing network |

## 5.1 Top Level

Top level is the three cells `maspan`, `panic_top` and `panic`. These will be explained in reverse order. The reason for dividing the top level into these three cells, as opposed to including everything into the `panic` schematic, is convenience. Using this hierarchy the layout becomes more manageable.

### panic (Schematic 6 - Schematic 10)

The cell `panic` is the "logical" top level of PAnIC. It contains everything that makes PAnIC work (except output buffers).

### panic_top (Schematic 2)

The cell `panic_top` is `panic` plus output buffers. The `outputbuff` cells are there to drive the analog io pads. The `panic_io_buff` is there to convert the `panic_io` (input/output) into `panic_in` (input) and `panic_out` (output). This operation is performed since the digital IO pads (IOF3) take 1 input and 1 output signal, not an input/output signal. The `cb_100u` cell provides bias currents for the `outputbuff` cells.

### maspan (Schematic 1)

The cell `maspan` (master panic) is `panic_top` plus pads as shown in the schematic. Further description of the pads can be found at AMS homepage [18].

## 5.2 Panic_Control

`Panic_Control` (Schematic 12) consists of address register, table of content, input/output fallback, 4 to 16 decoder, serial peripheral interface and control. These modules will be explained separately.

### Address register (Schematic 18)

The address register holds the module and line address. It is an 8-bit synchronous register with `clk` connected to the clock input. The three least significant bits are connected to the `lineAddr` and bits 3 to 6 connected to `modAddr` (bit 7 is not used). A positive transition on `clk` will load the register with the value on `dBusIn`. Setting reset signal high will reset the register to 0x00.

### Table of Content (Schematic 19)

The `TOC` was written in VHDL as a read only memory (ROM) with a 16 x (2 x 8) structure, in other words the `modAddr` selects a (2 x 8) ROM and `lineAddr` selects the byte to be read. This way there is no need for an internal address register in the ROM,

it can use the existing address register. The `TOC` stores the cell number at line address 0 and the sub number at line address 1. During synthesis it was left to Synopsys to synthesize the most efficient structure. A positive transition on `tocwritetobus` will write the current value to `dBusOut`.

Table 17 shows the content of the TOC at the different addresses. The sub numbers may seem meaningless, but they are not. As mentioned before the first number (line address "000") defines the class of circuit, and the sub number defines a specification. Since this is the first IC based on the PAnIC architecture the sub numbers equal "0000 0001". If for example the two 8-bit DACs were different i.e. based on different topologies, the sub numbers for the DACs would be different.

*Table 17 TOC content*

| address | content | description |
|---|---|---|
| 00001 000 | 0000 0001 | sample and hold |
| 00001 001 | 0000 0001 | sub nr |
| 00010 000 | 0000 0010 | Comparator |
| 00010 001 | 0000 0001 | sub nr |
| 00011 000 | 0000 0011 | Bandgap voltage reference |
| 00011 001 | 0000 0001 | sub nr |
| 00100 000 | 0000 0100 | 8-bit DAC |
| 00100 001 | 0000 0001 | sub nr |
| 00101 000 | 0000 0100 | 8-bit DAC |
| 00101 001 | 0000 0001 | sub nr |

## IoFallBack (Schematic 20)

This is the secondary input/output interface, it consists of an 8-bit multiplexer and tristate buffers. Table 18 describes the `IoFallBack` states.

*Table 18 IoFallBack states*

| cntr_io | Disable_SPI | panic_io | to_modules |
|---|---|---|---|
| 00 | 0 | from_irs | high impedance |
| 01 | 0 | from_toc | high impedance |
| 1X | 1 | input | panic_io |

## Decode4_16 (Schematic 21)

Decode4_16 is a 4 to 16 decoder with enable. It translates the module address into enable signals for the analog module.

## Serial Peripheral Interface (Schematic 22)

The SPI is the primary data communication interface of the PAnIC. The behavior of the SPI was written in VHDL, it was left to the Synopsys to synthesize the most efficient structure. Figure 9 shows the timing diagram for one data transmission. The signals ss, sck and mosi (master out, slave in) are from the microcontroller. The micro-controller tells PAnIC that a data transmission is about to start by setting ss low. The value the micro-controller wants to transmit is "11110000", the current value of the SPI is "00001111". When sck goes high the SPI sets miso (master in, slave out) to the MSB and loads LSB with the value of mosi. After the transmission the value on dBusOut will be "11110000" while the value transmitted to the micro-controller will be "00001111". Table 19 lists the SPI states.

*Table 19 SPI states*

| readTOC | readIRS | enable | Disable_SPI | dBusOut | SPI value |
|---|---|---|---|---|---|
| X | X | X | 1 | high impedance | no change |
| X | X | 0 | 0 | high impedance | no change |
| X | X | 1 | 0 | SPI value | no change |
| 1 | 0 | 1 | 0 | SPI value | dBusTOCIn |
| 0 | 1 | 1 | 0 | SPI value | dBusIn |

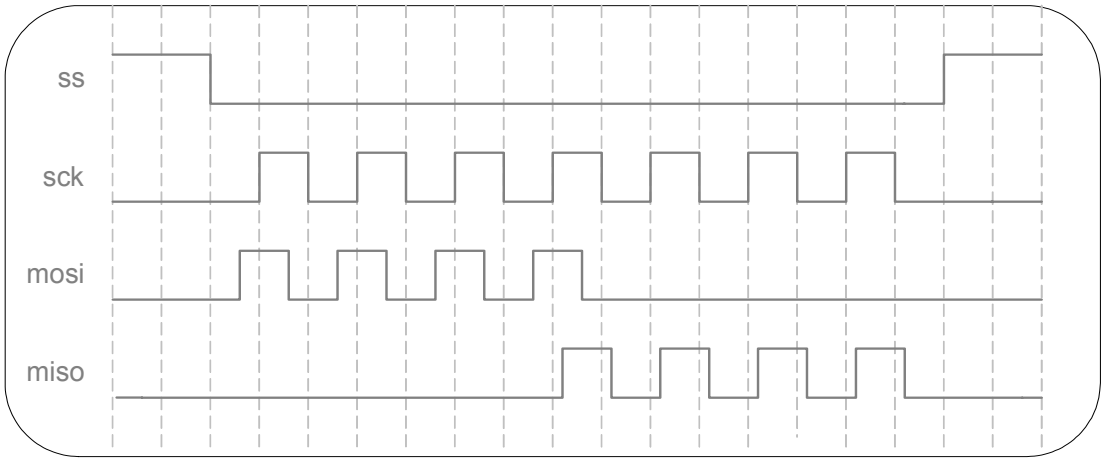



*Figure 9 Timing diagram for the SPI module*

### Control (Schematic 23)

`Control` decodes the `cntr` into control signals for different portions of PAnIC. Table 20 lists the different states depending on `enable` and `cntr`.

*Table 20 Control states*

| enable | cntr | addreadfrombus | modreadfrombus | modwritetobus | tocwritetobus |
|---|---|---|---|---|---|
| 0 | X | 0 | 0 | 0 | 0 |
| 1 | 000 | 0 | 0 | 0 | 0 |
| 1 | 001 | 1 | 0 | 0 | 0 |
| 1 | 010 | 0 | 1 | 0 | 0 |
| 1 | 011 | 0 | 0 | 1 | 0 |
| 1 | 100 | 0 | 0 | 0 | 1 |
| 1 | 101 | 0 | 0 | 0 | 0 |
| 1 | 110 | 0 | 0 | 0 | 0 |
| 1 | 111 | 0 | 0 | 0 | 0 |

## 5.3 Analog Module Framework

The AMF comes in two flavors; `AMF_LMIR2OR2` (Schematic 13) and `AMF_LMOR2` (Schematic 14). Only the contents of `AMF_LMIR2OR2` will be explained since `AMF_LMOR2` is an `AMF_LMIR2OR2` without the input registers. The `AMF_LMIR2OR2` contains an `AMF_Digital_LMIR2OR2`, two `IRS_analog` and two `ORS_analog` blocks.

### AMF_Digital_LMIR2OR2 (Schematic 24)

The `AMF_Digital_LMIR2OR2` consist of `LineDec`, `ModReg` , `IRS_Digital` and `ORS_Digital`,

#### *LineDec (Schematic 28)*

`LineDec` is a 3 to 8 decoder. It provides enable signals for the `ModReg`, `IRS_Digital` and `ORS_Digital` cells.

#### *ModReg (Schematic 31)*

`ModReg` is an 8-bit synchronous register that provides a digital input for the analog cell. It loads the value on `dBusIn` if enable is high and a positive transition occurs on `modreadfrombus`. It was originally intended to provide a digital output as well, but because of an error in the VHDL code, which was discovered late in the design process, it can only write the current value of the register to `dBusOut`. Since none of the analog cells use a digital output this error was not corrected.

### *IRS_Digital (Schematic 29)*

IRS_Digital is the digital portion of the IRS, it provides control signals (acontrol) for analog switches in IRS_analogv2. It loads the value on dBusIn if enable is high, reset is low and a positive transition on modreadfrombus occurs. The port irsloadanalog is connected to the modwritetobus signal from Control. When irsloadanalog is high the value of acontrol is written to dBusOut. The register is reset to 0x00 when the reset signal is high. Table 21 describes the IRS_Digital states. The irsloadanalog signal is connected to the modwritetobus signal from Control in Panic_Control.

*Table 21 IRS_Digital states*

| **enable** | **modreadfrombus** | **irsloadanalog** | **dBusOut** | **register value** |
|---|---|---|---|---|
| 0 | X | X | high impedance | no change |
| 1 | 1 | 0 | high impedance | dBusIn |
| 1 | 0 | 1 | acontrol | no change |

### *ORS_Digital (Schematic 30)*

ORS_Digital is the digital portion of ORS, it provides control signals (ors_control(6:0)) to ORS_analog. It loads the value on dBusIn if enable is high and a positive transition occurs on modreadfrombus. The register is reset to "000" if reset is high. It provides control signals for ORS_analog switches. Table 22 describes the ORS_digital states.

*Table 22 ORS_Digital states*

| **register value** | **ors_control** | **state** |
|---|---|---|
| 000 | 000 0001 | normal |
| 001 | 000 0011 | aGlobOut0 |
| 010 | 000 0101 | aGlobOut1 |
| 011 | 000 1001 | aGlobOut2 |
| 100 | 001 0001 | aGlobOut3 |
| 101 | 010 0000 | Low |
| 110 | 100 0000 | High |
| 111 | 000 0001 | normal |

## IRS_analogv2 (Schematic 26)

IRS_analogv2 is the analog portion of the IRS and contains eight irs_cellsv2. Table 23 describes the IRS_analogv2 states.

The irs_cellv2 is shown in Schematic 32. Transistors m_1, m_2 and m_3, m_4 are inverters that invert the wtb signal (modwritetobus from Control) into wtb_inv and a delayed wtb. The transistor m_5 is the analog switch; we have not used a transmissiongate here because the signal swing of the input signal is 0-3V while the supply voltage of the irs_cell is 5V. The transistors m_7, m_8 and m_9, m_10 are transmissiongates. When wtb is low m_9, m_10 is on and acontrol is connected to the

gate of m_5. When wtb is high m_7, m_8 is on and in is connected to acontrol. The transistor m_11 is a pull-down transistor for the gate of m_5 so leakage current trough m_4 or m_9, m_10 does not turn m_5 on during readback. The control signal for m_11 (wtb2) is delayed so switching currents through m_11 is minimized when switching loadirsanalog from low to high. loadirsanalog is connected to the modwritetobus signal from control in Panic_Control

*Table 23 IRS_analogv2 states*

| **loadirsanalog** | **acontrol** |
|---|---|
| 0 | input from IRS_Digital |
| 1 | in(7:0) |

## ORS_analog (Schematic 25)

ORS_analog is the analog portion of the ORS. It has 7 different states depending on the value of the ors_control(6:0) bus from ORS_Digital. Table 24 shows the outputs of ORS_analog in the different states. From the schematic we se that ors_control(4:0) control switches while ors_control(5) and ors_control(6) control pull-down and pull-up transistors. The pull-up and pull-down are used during ReadBack to provide high and low states. The cell called buff_ors is a buffer for driving the routing network.

Since aGlobOut0-3 are signals with several ORS_analog blocks connected to them, there is a possibility that two (or more) signals might be shorted. The layout of the ORS_analog cell and the aGlobOut0-3 nets allow such a mistake up to a certain point. It can tolerate up to five ORS_analog blocks driving a aGlobOut signal high while one ORS_analog block driving the signal low, in other words, the buffer and surrounding nets can tolerate five times the nominal DC current and still operate within limits.

*Table 24 ORS_analog states*

| **Ors_control** | **State** | **aGO0** | **aGO1** | **aGO2** | **aGO3** | **Output** |
|---|---|---|---|---|---|---|
| 000 0001 | Normal | | | | | In |
| 000 0011 | Glob0 | In | | | | In |
| 000 0101 | Glob1 | | In | | | In |
| 000 1001 | Glob2 | | | In | | In |
| 001 0001 | Glob3 | | | | In | In |
| 010 0000 | Low | | | | | Low |
| 100 0000 | High | | | | | High |
| 000 0001 | Normal | | | | | In |

### buff_ors (Schematic 33)

This is a buffer based on a two-stage operational amplifier with miller compensation.

## 5.4 Support circuitry

Supporting the main cells there are some cells that will be described here.

### outputbuff (Schematic 4)

This is a buffer based on a two-stage operational amplifier with miller compensation. It is used to drive the `aGlobOut0-3` pins.

### cb_100u (Schematic 5)

This is a current copier for supplying 100µA bias currents to the output buffers.

### panic_io_buff (Schematic 3)

Since the digital io pads requires one input and one output signal this cell was created to separate the `panic_io(7:0)` bus into `panic_in(7:0)` and `panic_out(7:0)`. It consists of 8 `panic_io_tribuff` cells (Schematic 11).

### curr_combine (Schematic 15)

This cell divides one 10µA current into two 5µA currents, used to bias the buffers in `ORS_analog` cells.

### curr_combine_cells (Schematic 16)

This cell divides one 100µA current into 4 5µA currents, used to bias the sample & hold, differential buffer, bandgap voltage reference and comparator.

### curr_combine10u (Schematic 17)

This cell divides one 100µA current into six 10µA currents. It is used to bias the `curr_combine` cells.

# 6. PAnIC layout

Figure 12, on page 38, shows the finished layout of PAnIC, PAnIC is 3.8 mm wide and 2.9 mm high. The final layout is included on the CD-ROM as a GDS-II file.

As always, there are many considerations when performing layout of a mixed-signal design, i.e. power stability, substrate noise, crosstalk and matching of devices. Special care has been given to two of these; power stability and substrate noise, as specified. Another important consideration was to provide enough current to the cells. The calculations and dimensioning of the power nets will be explained first, secondly how to make stable power and thirdly an explanation of guard-rings.

## 6.1 Dimensioning power

Simulations in the synthesis tool and manual calculations have been performed to make sure that the power lines driving the different cells can handle the current that the cell needs. In the process the dc current pr µm metal was 1mA and a via could handle 0.5mA. We started by calculating the power used in the digital portions of PAnIC when operating at a frequency of 10MHz, which is 10 times the intended operating frequency. The calculations are shown in Table 25, Table 26 and Table 27. Table 28 shows the total current estimation for all cells in PAnIC. Table 29 shows the power nets and the final dimension for each net

The digital cells are all made from standard cells in the digital standard cell library of the foundry.

*Table 25 Power used in AMF cells*

| **Std. cells** | **µW/MHz** | **LineDec** | | **ModReg** | | **ORS_Digital** | | **IRS_Digital** | |
|---|---|---|---|---|---|---|---|---|---|
| | | Nr | µW/MHz | Nr | µW/MHz | Nr | µW/MHx | Nr | µW/MHz |
| AN21 | 4,9 | 0 | 0 | | 0 | | 0 | 1 | 4,9 |
| NA3 | 4,27 | 0 | 0 | | 0 | | 0 | 1 | 4,27 |
| NO2 | 4,15 | 0 | 0 | | 0 | 5 | 20,75 | | 0 |
| AND4 | 6,13 | 0 | 0 | | 0 | 1 | 6,13 | | 0 |
| OR2 | 4,85 | 0 | 0 | | 0 | 1 | 4,85 | 1 | 4,85 |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| EN1 | 4,79 | 0 | 0 | | 0 | 1 | 4,79 | | 0 |
| DFA | 9,7 | 0 | 0 | 8 | 77,6 | 3 | 29,1 | 8 | 77,6 |
| IT2 | 4,52 | 0 | 0 | 8 | 36,16 | | 0 | 16 | 72,32 |
| AND2 | 5,12 | 0 | 0 | 1 | 5,12 | 1 | 5,12 | 1 | 5,12 |
| NO3 | 4,51 | 8 | 36,08 | | 0 | | 0 | | 0 |
| NA2 | 3,62 | 2 | 7,24 | 1 | 3,62 | 3 | 10,86 | 8 | 28,96 |
| IN1 | 3,11 | 3 | 9,33 | 1 | 3,11 | 1 | 3,11 | 10 | 31,1 |
| | | **13** | **52,65** | **19** | **125,6** | **16** | **84,71** | **46** | **229,1** |

*Table 26 Power and current used in AMF*

| **Cell** | **µW/MHz** | **AMF_Digital_LMOR2** | | **AMF_Digital_LMIR2OR2** | |
|---|---|---|---|---|---|
| | | Nr | µW/MHz | Nr | µW/MHz |
| LineDec | 52,65 | 1 | 52,65 | 1 | 52,65 |
| ModReg | 125,61 | 1 | 125,6 | 1 | 125,6 |
| Ors_Digital | 84,71 | 2 | 169,4 | 2 | 169,4 |
| IRS_digital | 229,12 | | 0 | 2 | 458,2 |
| | | **4** | **347,7** | **6** | **805,9** |
| | | | | | |
| **Frequency** | **Supply** | | **Current (mA)** | | **Current(mA)** |
| **10MHz** | **5V** | | **0,69536** | | **1,61184** |

*Table 27 Power and current used in Panic_Control and sub-cells*

| **Std. C.** | **µW/MHz** | **AddrReg** | | **Control** | | **Decode4_16** | | **TOC** | | **SPI** | | **IoFallBack** | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| AN21 | 4,9 | 0 | 0 | | 0 | | 0 | | 0 | 8 | 39,2 | | 0 |
| NA3 | 4,27 | 0 | 0 | | 0 | 2 | 8,54 | | 0 | | 0 | | 0 |
| NO2 | 4,15 | 0 | 0 | 1 | 4,15 | 17 | 70,55 | | 0 | 2 | 8,3 | | 0 |
| AND4 | 6,13 | 0 | 0 | | 0 | | 0 | | 0 | | 0 | | 0 |
| OR2 | 4,85 | 0 | 0 | | 0 | | 0 | | 0 | 1 | 4,85 | | 0 |
| OR3 | | | | 1 | | | | 1 | | | 0 | | 0 |
| EN1 | 4,79 | 0 | 0 | | 0 | | 0 | | 0 | | 0 | | 0 |
| DFA | 9,7 | 7 | 67,9 | | 0 | | 0 | | 0 | | 0 | | 0 |
| IT2 | 4,52 | 0 | 0 | | 0 | | 0 | | 0 | 8 | 36,16 | 17 | 76,84 |
| AND2 | 5,12 | 0 | 0 | | 0 | | 0 | | 0 | 1 | 5,12 | | 0 |
| NO3 | 4,51 | 0 | 0 | | 0 | | 0 | 1 | 4,51 | | 0 | | 0 |
| NA2 | 3,62 | 0 | 0 | | 0 | 6 | 21,72 | 2 | 7,24 | 18 | 65,16 | | 0 |
| IN1 | 3,11 | 1 | 3,11 | 3 | 9,33 | 4 | 12,44 | 3 | 9,33 | 26 | 80,86 | 18 | 55,98 |
| NO4 | 6,91 | | 0 | 1 | 6,91 | | 0 | 1 | 6,91 | | 0 | | 0 |
| AND43 | 13,54 | | 0 | 1 | 13,54 | | 0 | | 0 | | 0 | | 0 |
| NO23 | 12,68 | | 0 | 1 | 12,68 | | 0 | | 0 | | 0 | | 0 |
| MU2 | 5,45 | | 0 | | 0 | | 0 | 2 | 10,9 | 25 | 136,25 | 8 | 43,6 |
| DF8 | 8,94 | | 0 | | 0 | | 0 | 7 | 62,58 | | 0 | | 0 |
| DFS8 | 9,96 | | 0 | | 0 | | 0 | 1 | 9,96 | | 0 | | 0 |
| LOGIC0 | | | 0 | | 0 | | 0 | 1 | 0 | | 0 | | 0 |
| EN1 | 4,8 | | 0 | | 0 | | 0 | 1 | 4,8 | | 0 | | 0 |
| NA22 | 6,77 | | 0 | | 0 | | 0 | | 0 | 1 | 6,77 | | 0 |
| ON221 | 6,76 | | 0 | | 0 | | 0 | | 0 | 8 | 54,08 | | 0 |
| ON211 | 3,96 | | 0 | | 0 | | 0 | | 0 | 8 | 31,68 | | 0 |

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DFB | 5,2 | | 0 | | 0 | | 0 | | 0 | 17 | 88,4 | | 0 |
| OR23 | 12,16 | | 0 | | 0 | | 0 | | 0 | 1 | 12,16 | | 0 |
| | | **8** | **71,01** | **8** | **46,61** | **29** | **113,25** | **20** | **116,23** | **124** | **568,99** | **43** | **176,42** |
| | | | | | | | | | | | | | |
| **PAnIC_Control** | | **1092,51 µW/MHz** | | | | **VDD: 5V** | | **Freq: 10MHz** | | | **Current: 2,18 mA** | | |

*Table 28 Current estimation for cells*

| **Cell** | **Estimated Current** | **Sized for** | **Supply Net** | **Ground Net** | **Nr of cells** | **Total Current (mA)** |
|---|---|---|---|---|---|---|
| AMF_Digital_LMOR2 | <1mA (10MHz) | 2mA | VDD | VSS | 3 | **6** |
| AMF_Digital_LMIR2OR2 | <2mA (10MHz) | 2mA | VDD | VSS | 3 | **6** |
| IRS_analogv2 | <1uA | 1mA | VDDA | VSSA | 6 | **6** |
| ORS_analog | 0,4mA | 2mA | VDDA | VSSA | 12 | **24** |
| SH_GR03 | 2.5mA | 3mA | VDDA3V | VSSA | 1 | **3** |
| OTA_GRP10 | <1mA | 1mA | VDDA3V | VSSA | 1 | **1** |
| DAC | <2mA | 4mA | VDDA | VSSA | 2 | **8** |
| BGR | < 0,1mA | 1mA | VDDA3V | VSSA | 1 | **1** |
| Comparator | < 2uA | 2mA | VDDA3V | VSSA | 1 | **2** |
| Panic_Control | <3mA (10MHz) | 4mA | VDD | VSS | 1 | **4** |

*Table 29 Size power nets*

| **Power net** | **Estimated current (mA)** | **Size(µm)** |
|---|---|---|
| VDD | 16 | 20 |
| VDDA5V | 38 | 40 |
| VDDA3V | 7 | 10 |
| VSS | 16 | 20 |
| VSSA | 46 | 50 |

## 6.2 Making stable power

All metal lines on an integrated circuit have a certain sheet resistance per square. If you do not take this into consideration when you route your power nets you will run into trouble when using the chip. Figure 10 shows an example of bad power routing, the block diagram on the left translates into the circuit diagram on the right. If the current drain from the cells is stable, the voltages V1, V2 and V3 will be stable but V3 < V2 < V1 < VDD because of the series resistance. We cannot get away from this series resistance between a pin and the cell, there will always be a difference between VDD and the supply voltage of the cell. We can minimize this resistance in two ways; shorter power nets or wider power nets.

The big problem starts when current drain from a cell varies, which it normally does. The voltages V1, V2 and V3 will change proportional to the change in the voltage drop over the series resistance. Table 30 describes calculations of voltage drops over resistances in Figure 10. Here the sheet resistance is 150 mΩ per square, the widths of the lines are 10 µm and VDD is 5V. In the first case V1=4.835V,

V2=4,715V and V2=4,715V. In the second case the DAC is switching and draws a 10mA current instead of the nominal 1mA current. In this case V1=4,565V, V2=4,243V and V3=3,838V. As we can see, the change in current in the DAC has large effects on the supply voltages; ΔV1 = 0,27V, ΔV2 = 0,472V and ΔV3 = 0.877V. A supply voltage change of almost 1V can have devastating effects on the performance of a cell. A similar resistor network will exist between VSS and ground of a cell, thus further decreasing the effective potential the cell has available.

To minimize voltage drop and voltage changes several strategies have been employed. The power nets are divided into analog and digital supply nets, these supply nets are connected to the VDD and VSS pads using wide metal lines to lower the sheet resistance. The power nets are VDD, VDDA VDDA3V, VSS and VSSA. These nets are routed, as we can see in Figure 12, in a large ring around PAnIC. It is connected to pads at three places, top, left and right. All cells in PAnIC draw power from the ring, thus minimizing voltage drop due to current changes in neighboring cells. All power lines to cells are routed in metal 3 since it has the lowest sheet resistance (half of metal 2). Cells with large current changes (such as the DAC and sample & hold) have been placed close to a pin.

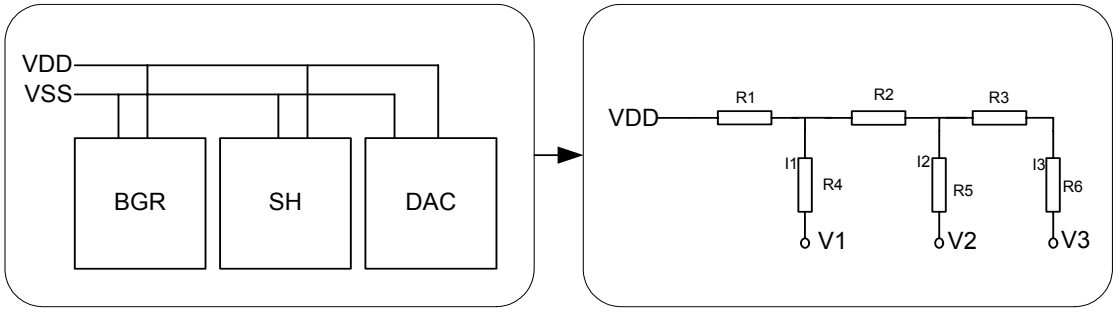



*Figure 10 Example power routing*

*Table 30 Power calculations on Figure 10*

| | Length | Resistance | Curr. Name | Current (mA) | Voltage Drop | Current (mA) | Voltage Drop |
|---|---|---|---|---|---|---|---|
| R1 | 2000 | 30 | I1+I2+I3 | 5 | 0,15 | 14 | 0,42 |
| R2 | 1500 | 22,5 | I2+I3 | 4 | 0,09 | 13 | 0,2925 |
| R3 | 2000 | 30 | I3 | 1 | 0,03 | 10 | 0,3 |
| R4 | 1000 | 15 | I1 | 1 | 0,015 | 1 | 0,015 |
| R5 | 1000 | 15 | I2 | 3 | 0,045 | 3 | 0,045 |
| R6 | 1000 | 15 | I3 | 1 | 0,015 | 10 | 0,15 |

## 6.3 Guard-rings

Enclosing all analog cells in PAnIC we have a guard-ring that separates the digital substrate from analog substrate (Figure 12). A cross-section of the guard ring is shown in Figure 11. This guard ring decouples the digital (left side) and the analog (right side) via the capacitor between the N well and the P substrate. This prevents substrate noise propagation between digital and analog substrate.

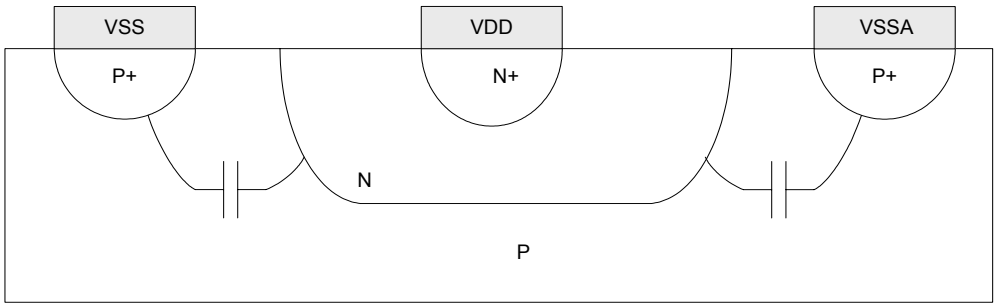



*Figure 11 Guard-ring cross-section*

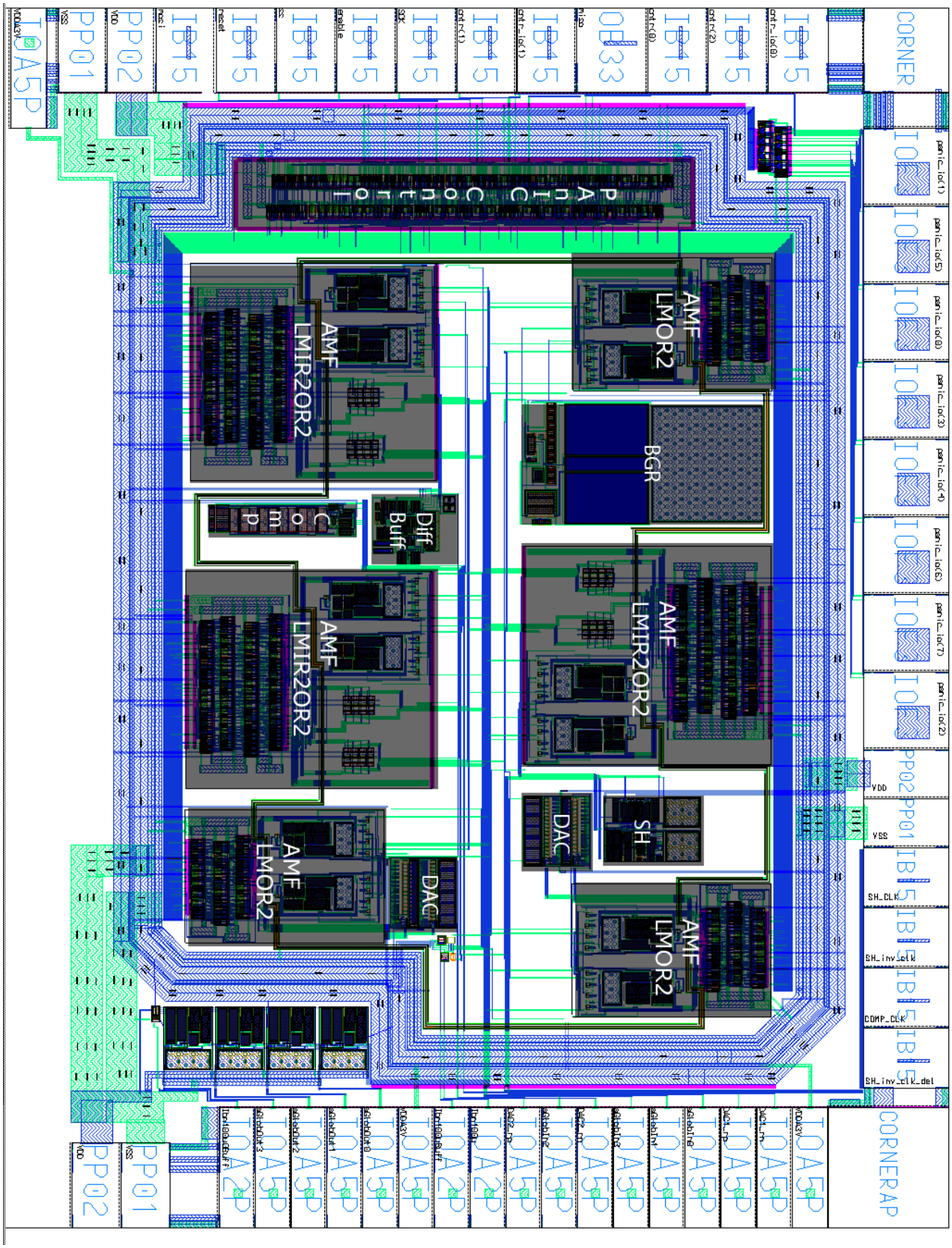



*Figure 12 PAnIC layout with pads*

# 7. Verification

## 7.1 Verification of digital functions

The verification of the digital functions was performed in ModelSim. The tests that were performed were test of the SPI interface test of reset, test of readback function, test of TOC, test of routing and test of the IoFallBack interface. The results from these test are not included in this report since the mixed-signal verification covers the same tests. The VHDL testbench can be found on the CD-ROM.

## 7.2 Verification of routing network

As mentioned earlier, the routing network consists of `irs_cellv2`, `ORS_analog` and `outputbuff`. Each of these cells were verified according to specification. The verification was performed using SPICE. The cells were simulated with typical, worst speed and worst power model parameters. All cells were found to operate within specified limits. Presented here are the results from the simulation.

### irs_cellv2

The `irs_cellv2` was simulated with a load of 3pF and a supply of 5V. The results from the simulations are shown in the table below.

*Table 31 irs_cellv2 results*

| Name | Best | Typical | Worst | [] |
|---|---|---|---|---|
| Signal swing | 0 - 4.1 | 0 - 3.8 | 0 – 3.6 | V |
| Bandwidth | 100 | 65 | 40 | MHz |
| Off dampning | < 90 | < 80 | < 75 | dB |
| Phase shift up to 1MHz | 0.15 | 0.23 | 0.35 | deg |
| Phase shift at 30MHz | 4.5 | 6.8 | 10.3 | deg |

## ORS_analog

The `ORS_analog` was simulated with a bias current of 5µA, 5V supply and 5pF load. The result from the simulation is shown in the table below.

*Table 32 ORS_analog results*

| Name | Best | Typical | Worst | [] |
|---|---|---|---|---|
| Signal swing | 0 – 4.1 | 0 – 3.8 | 0 – 3.6 | V |
| Bandwidth | 40 | 36 | 33 | MHz |
| Off dampning | 56 | 52 | 50 | dB |
| Phase shift up to 1MHz | 1.9 | 2.2 | 2.8 | deg |
| Phase shift at 30MHz | 70 | 82 | 95 | deg |
| Slewrate | 10 | 9.3 | 8.4 | V/µs |

## Outputbuff

The `outputbuff` was simulated with bias current of 100µA, supply of 5V and a 50pF load. The results from the simulations are shown in the table below.

*Table 33 outputbuff results*

| Name | Best | Typical | Worst | [] |
|---|---|---|---|---|
| Signal swing | | 0 – 4.2 | | V |
| Bandwidth | 90 | 85 | 80 | MHz |
| Phase shift up to 1MHz | 0.9 | 1.2 | 1.5 | deg |
| Phase shift at 30MHz | 30 | 35 | 37 | deg |
| Slewrate | 54 | 50 | 46 | V/µs |

## 7.3 Mixed-Signal Verification

Each mixed-signal cell has undergone a functional test using extracted netlist from the schematic. Due to problems with the design software we were never able to extract a netlist from the finished layout. Although this is a disadvantage we are confident that the simulations on netlist from schematics in combination with LVS is sufficient to verify the operation of PAnIC. More on this point will be discussed in the next chapter. The results presented here are simulations on the top-level netlist including pads (extracted from Schematic 1). The results from the lower levels in the hierarchy are not presented since they are implicit in the following results. The verification of the timing demands is included as part of the comparator simulation.

### Analog Cells

#### *DAC*

Figure 13 shows the simulation of `DAC1`, the simulation of `DAC2` has not been included since it is similar to the `DAC1` simulation. The output of `DAC1` is connected to `aGlobOut0`, this occurs at 3.5µs. The values "0000 0000", "0000 0111", "0000 1111", "0001 1111", "1111 1111" are then written to the `DAC1`. As we see the DAC responds and works as it should.

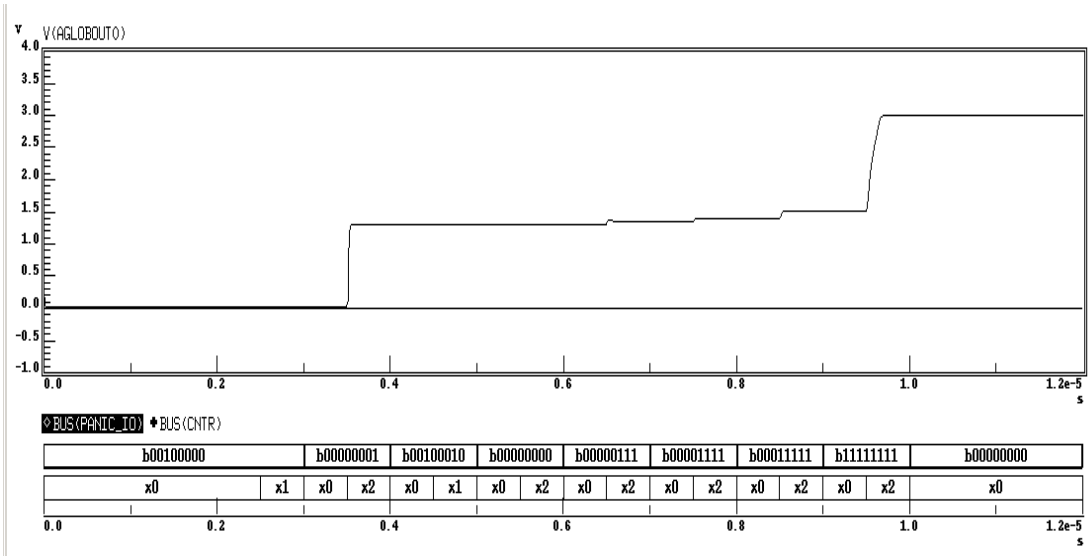



*Figure 13 DAC1 functional verification*

### *Bandgap Voltage Reference*

Figure 14 shows the verification of the Bandgap Voltage Reference. `Vref` is connected to `aGlobOut0` and `V_PTAT` is connected to `aGlobOut1`. At 3.0µs the `aGlobOut0` = 1.2V and `aGlobOut1` = 0.6V.

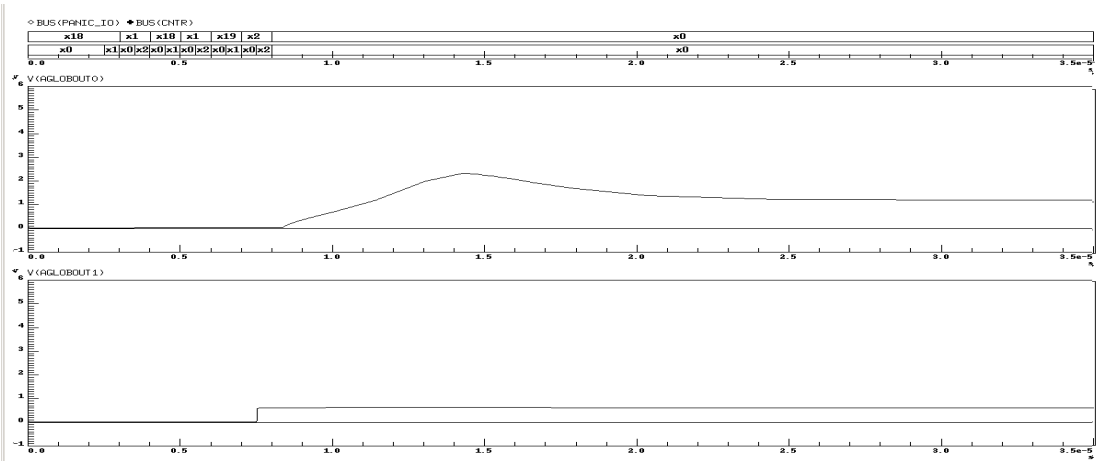



*Figure 14 Bandgap voltage reference functional verification*

### *Sample & Hold*

Figure 15 shows the result from the sample & hold simulation. `aGlobIn0` is connected to positive input and `aGlobIn1` is connected to negative input. `aGlobIn0` is ramped from 0V to 3V, `aGlobIn1` is ramped from 3V to 0V. `aGlobOut0` is connected to the positive output and `aGlobOut1` is connected to the negative output. The circuit samples when `sh_inv_clk` is high and holds when `sh_inv_clk` is low. As we can see from the result the sample & hold has an output swing of 1.3 - 3V.

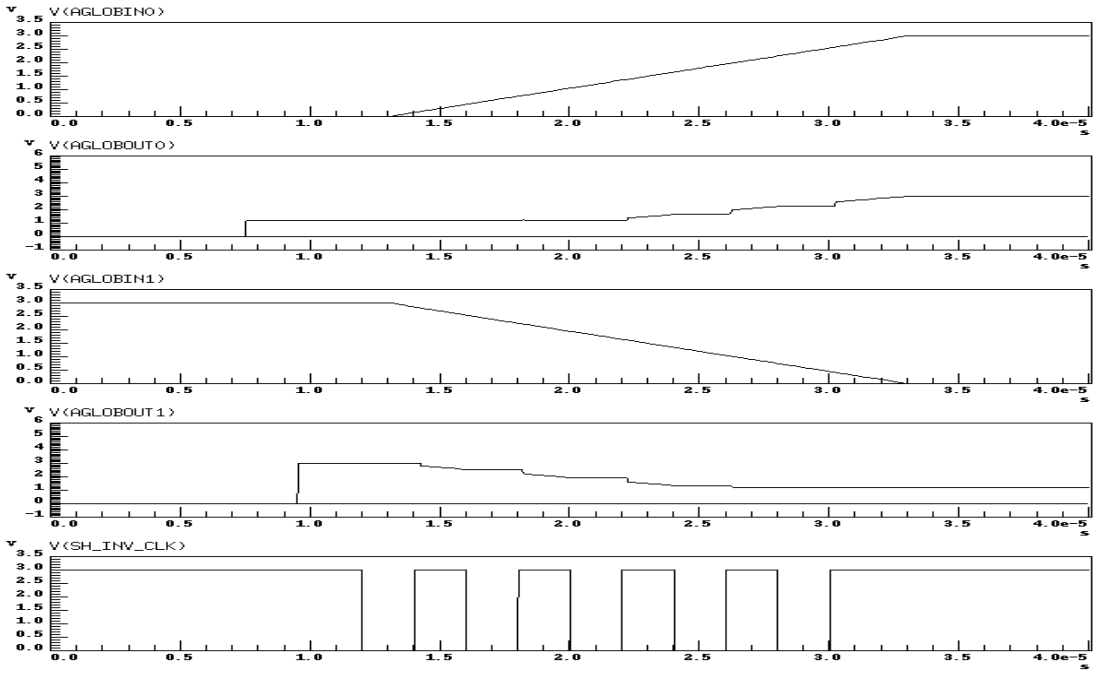



*Figure 15 Sample & hold functional verification*

### *Comparator*

Figure 16 shows the result from the comparator verification. `aGlobIn0` is connected to the positive input and `aGlobIn1` is connected to the negative input. `aGlobIn0` and `aGlobIn1` are switched between 1.5V and 1.508V. `aGlobOut0` is connected to the positive output and `aGlobOut1` is connected to the negative output. When `comp_clk` goes high the comparator samples the input signals. From `comp_clk` goes high till the output signal is valid there is a delay of 0.5µs. This translates to a clock frequency of 1MHz, as specified. The main limiter on the timing demand is the slew-rate of the internal buffers (`buff_ors`) cells.

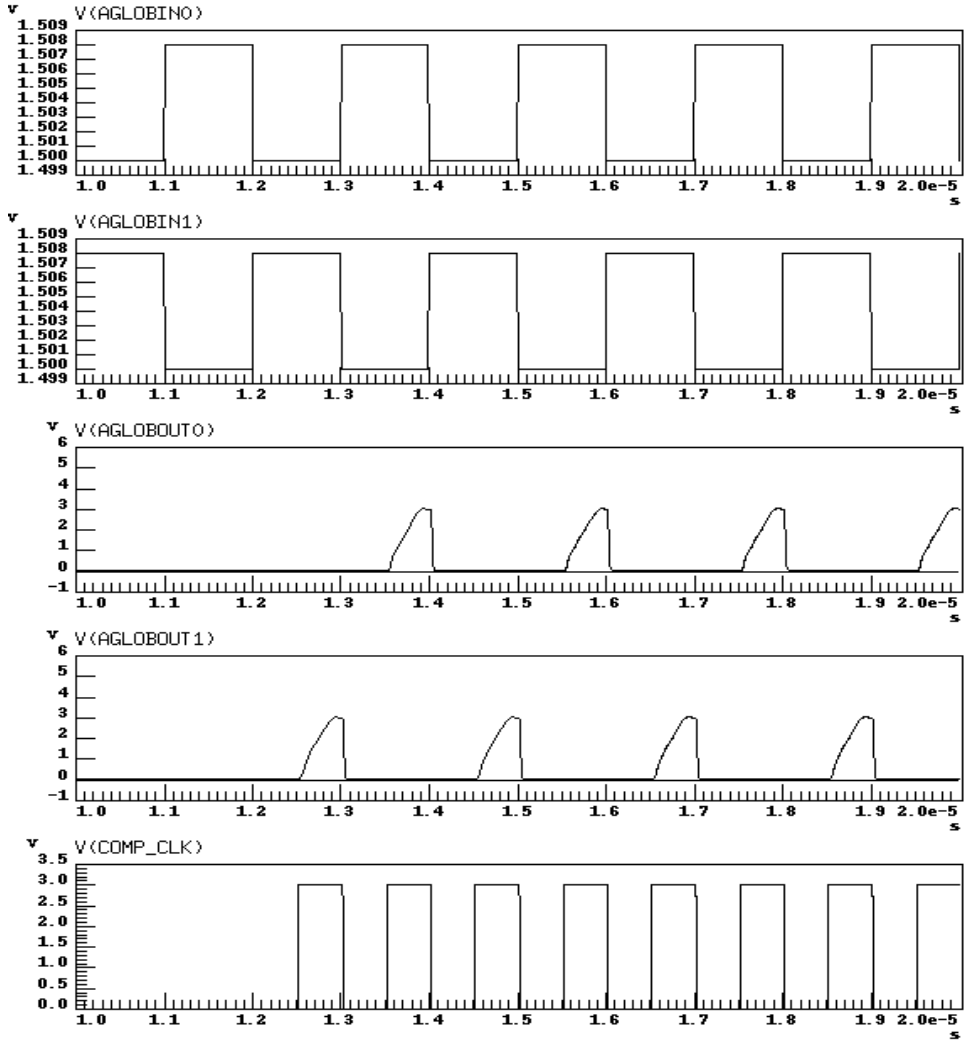



*Figure 16 Comparator functional verification*

### *Differential Buffer*

Figure 17 shows the result from the differential buffer verification. `vin_p` is connected to `aGlobIn0` and `vin_n` is connected to aGlobIn1. `vout_p` is connected to `aGlobOut2`. The output signal is valid from 14µs, the output signal has a peek to peek value of 0.4V.

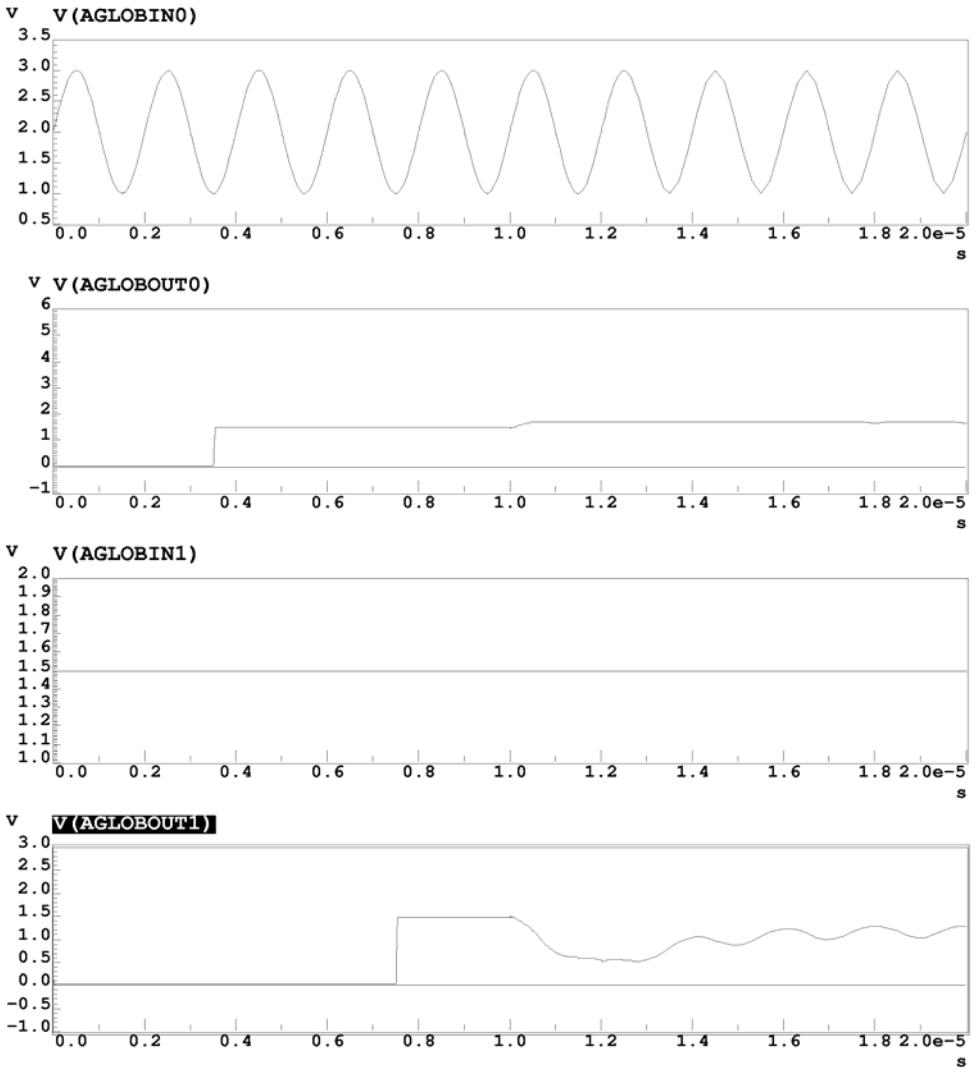



*Figure 17 Differential buffer functional verification*

## Analog to Digital converter verification

The ADC is connected as shown in Figure 8 (under Using PAnIC in the Design chapter). Figure 18 shows an excerpt from the ADC8 simulation. `DAC2` is set too "0000 1111", `DAC1` is switched between the values "0000 1110" (x0E) and "0001 0000" (x10). The positive output from the comparator is connected too `aGlobOut0`, and the negative is connected to `aGlobOut1`. The SH holds when `sh_inv_clk` is low. The comparator samples when `comp_clk` goes high. The first value is written to `DAC1` at 38.5µs (`cntr` = x2 = "010"), the SH starts holding at 40.3µs, the comparator starts comparing at 40.8µs. As we can see, the comparator output indicates that `DAC2` output signal is larger than the `DAC1` output signal, which is what we would expect. The same sequence repeats for the second value, and the comparator indicates that the current `DAC1` output signal is larger than the `DAC2` output signal.

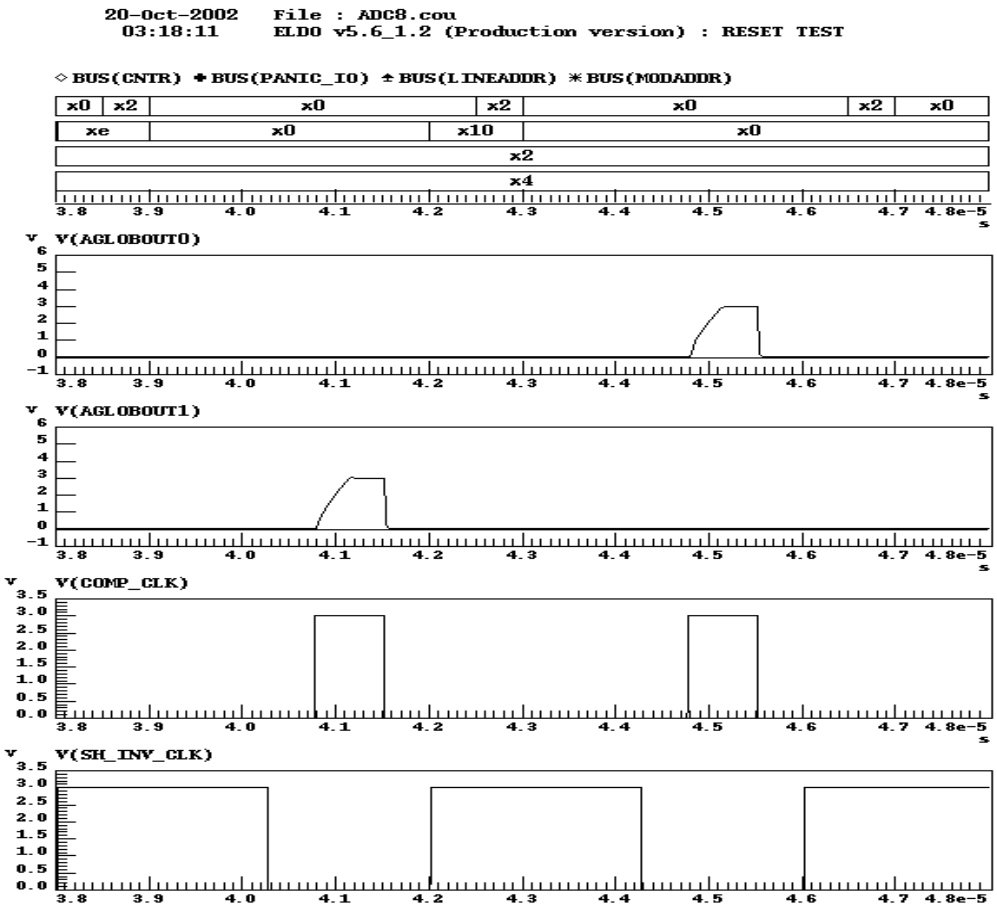



*Figure 18 ADC8 functional verification*

## TOC verification

Figure 19 shows an excerpt from TOC simulation. The `panic_io` bus from PAnIC is split into a `panic_in` and `panic_out` to make it possible to simulate in SPICE, this conversion is done outside the `maspan` cell. At 0.8µs the panic_in is set to "00011 000" (x18) at 8.5µs this is loaded into the address register (cntr= x1 = 001). At 9.5µs the `tocwritetobus` signal is given (`cntr` = x4 = 100). The output changes to "0000 0100" (x4). If we compare this to the TOC content (Table 17) we see that the result is correct. The same sequence repeats for the addresses "000100 000" and "00101 000".

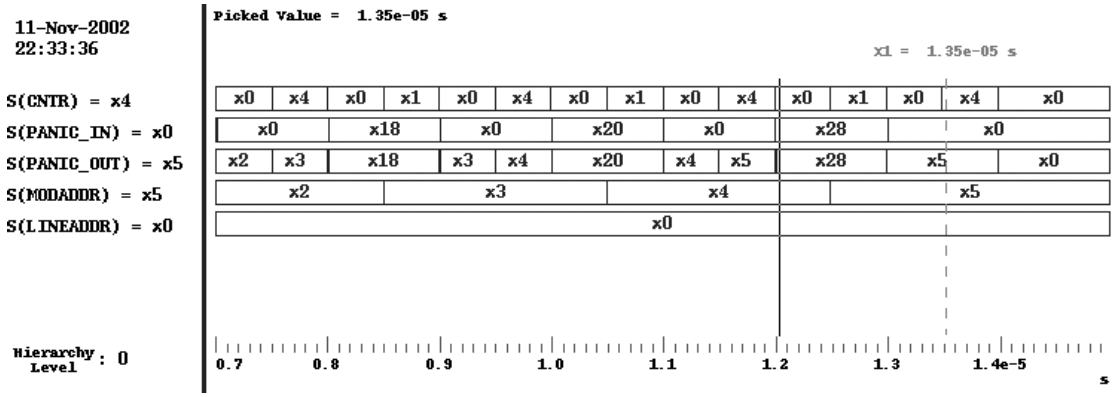



*Figure 19 Excerpt from TOC functional verification*

## ReadBack verification

The ReadBack verification was divided into three simulations, ReadBack for `COMP`, ReadBack for `DiffBuff` and ReadBack for `SH`. The reason for separating was that Eldo crashed when trying to run a complete ReadBack simulation. The signals for ReadBack were generated with a Perl script (included on the CDROM). It follows the pseudo code presented in the introduction (section 1.8). It starts by setting all ORS low, and then performs the ReadBack loop. The complete result from the ReadBack simulation is extensive and will not be presented here. An excerpt from the comparator simulation is shown in Figure 20. At 90.5µs it loads the address for DAC1, at 91.5µs it sets ORS0 of DAC1 to high, at 92.5µs it loads the address for IRS0 of the comparator and at 93.5µs it performs a ReadBack (cntr = 011). The panic_out value at this point changes to "0001 0000". If we compare this value to line 16 in Table 15, we see that it is indeed the DAC1 that should be connected to input 5 on IRS0 of the comparator.

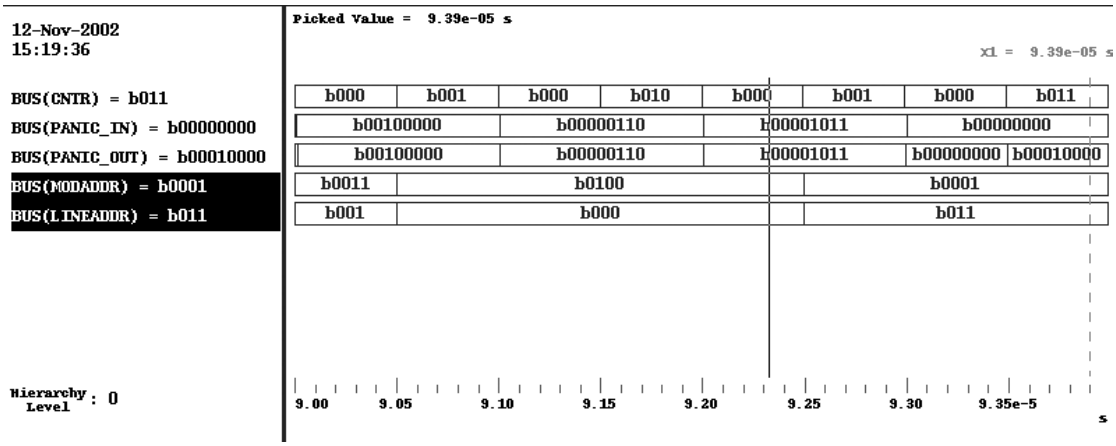



*Figure 20 Excerpt from readback of comparator functional verification*

## Data Interfaces

The `IoFallBack` data interface has been implicitly tested through the TOC and ReadBack verification. An excerpt from the SPI simulation is shown in Figure 21. The BUS(SPI) is the `ss` (LSB), `sck`, `mosi` (MSB) signals. The simulation shows a similar sequence as Figure 9 Timing diagram for the SPI module. `dBusOut` is read directly from the output of the `SPI` cell.

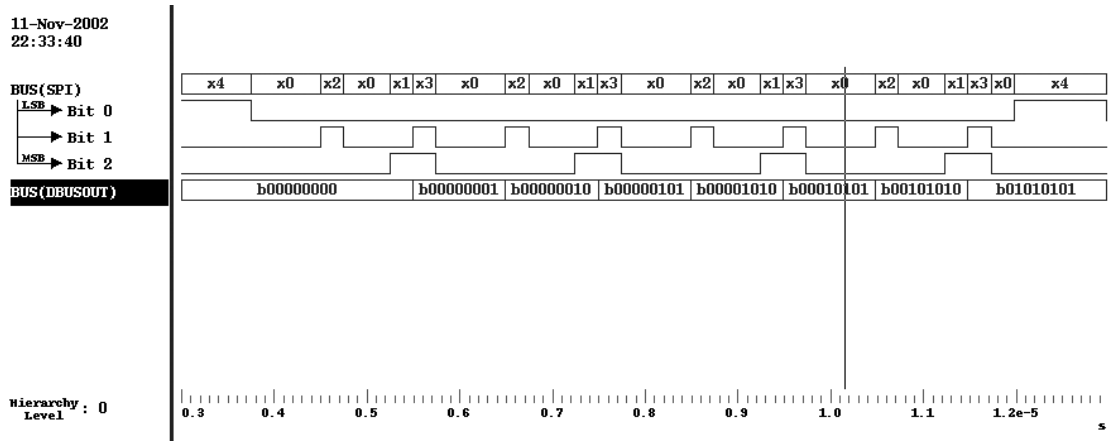



*Figure 21 Excerpt from SPI functional verification*

## 7.4 Layout verification

### Design Rule Check (DRC)

The design rule check has been performed. There are still some errors in the design, but most of these are IC station inability to recognize the power nets. The rest are DRC errors in the pads. The pads are made by the foundry and are therefore considered their responsibility. We remain confident that no critical DRC errors are present in the finished layout.

### Layout VS Schematic (LVS)

Layout VS Schematic has been performed on all cells in PAnIC. The top-level LVS can be found in Appendix IV. The LVS contains no errors, but two warnings. The warnings are:

- Unbalanced smashed mosfets were matched.
- Ambiguity points were found and resolved arbitrarily.

The unbalanced smashed mosfets are in the Differential Buffer and are considered to be correct. The ambiguity points are the two unconnected bias currents on `cb_100u` cell and are of no consequence.

# 8. Discussion & future work

Many issues have already been discussed in previous chapters, this chapter discusses some deviations from specification and verification plan and describes future work..

Creating PAnIC has been a more challenging project than we at first expected, and at times it has deserved its name. Due to problems with software tools the PAnIC tape-out has been postponed two times, but we finally have a product that we are confidant will work.

## 8.1 Area

A point where the specification has not been met is the chip area. We originally expected that we could create PAnIC on an area less than 10 square mm, the minimal area of the MPW run, but this turned out to be quite difficult. The total chip area is actually 4.0x3.8mm, were 11 square mm are the PAnIC, since 3 more independent circuits were included in the run.

## 8.2 Netlist extraction from layout

As common practice in ASIC design we indented to run the final verification of the PAnIC on an extracted netlist from layout. Unfortunately we were not able to do this due to problems with the design software. The first problem was that IC station did not extract devices with the correct model. The transistor models were easy to fix, but the extracted capacitor and resistor models were not. A second problem was that hierarchical extraction, which would have made changing the capacitor and resistor modules easier, extracted some of the digital cells twice and thereby produced a netlist that was incorrect. A third problem was that manually changing the capacitor and resistor models in a flat netlist could introduce errors. We decided that verification of schematic and verification of layout with LVS was sufficient to ensure the operation of PAnIC since all cells have been designed to cope with parasitic capacitances.

## 8.3 Near Future

The PAnIC chip is expected back from the foundry March of 2003. The creation of a test PCB is planned for the months January and February. If all goes well, a prototype remote laboratory using PAnIC should be finished at the end of summer 2003.

## 8.4 Next generation of PAnIC

Some observations have been made concerning future generations of the PAnIC chip. As it stands today, the PAnIC architecture is to complex to use an "IP" module for providing programmability to analog cells. Imagine that you have 16 analog cells which you want to be able to select through the same interface, and which you would like to be able to connect together. An IP version of PAnIC could do this with a small addition to the layout time. There are a couple things that need to be looked at to make this possible. First, the frameworks take up rather large area; the size of an `AMF_LMIR2OR2` makes the IRS ability to switch 8 inputs in any order an improbable feature to use. Using three AMFs to connect the circuit in Figure 1 (page 2) is, although possible, not that attractive since the connected circuit would probably be under 1% of the used area. A more attractive scenario is the way the AMFs has been used in PAnIC, connecting larger modules together. The IRS could therefore, in future versions; reduce the switching capability by only allowing 1 connection at a time, thus saving area. Another area saver would be to custom design the `IRS_Digital` and `ORS_Digital` cells on a transistor level and put more of the logic in `Panic_Control`. The goal should be to minimize area and simplifying addition of programmability during layout. It would also be advantageous to allow each analog cell to have a different set of IRS, ORS and ModReg modules.

The design of the `TOC` should be revisited in future versions. As it is today, changing the content of the `TOC` involves rewriting the VHDL code. A possible solution would be to separate the `TOC` from `Panic_Control`.

Early in the design phase (late summer of 2001) we considered using an on-chip microcontroller, but decided against due to complexity. This decision should be revisited in future versions.

# 9. Conclusion

The specification, design, implementation and verification of a programmable analog integrated circuit (PAnIC) with table of content has been presented. The architecture of PAnIC has been explained and proven through simulations to be a self-consistent programmable analog integrated circuit. The PAnIC offers extended flexibility, through circuit programmability, to our remote laboratory concept [1].

# 10. References

[1] Wulff, C. Ytterdal, T. Sæthre, T.A. Skjevlan, A. Fjeldy, T.A. *et al* "Next Generation Lab – A solution for remote characterization of analog integrated circuits". *International Caracas Conference on Devices, Circuits and Systems (ICCDCS-2002).* Found in Appendix VI and on the CDROM

[2] SystemC, http://www.systemc.org

[3] Wulff, C. Project report in SIE4094 "Programmable Analog Integrated Circuit w/table of content", Fall 2001 Department of Physical Electronics, Norwegian University of Science and Technology. Found on the CDROM

[4] Shen, H. Xu, Z. Dalager, B. Kristiansen, V. Strøm, Ø. et al "Conducting Laboratory Experiments over the Internet", *IEEE Trans. on Education, 42*, No. 3, 1999, pp. 180-18.

[5] Fjeldly, T.A. Shur, M.S. Shen, H. Ytterdal, T., "Automated Internet Measurement Laboratory (AIM-Lab) for Engineering Education", *Proceedings of 1999 Frontiers in Education Conference (FIE'99), San Juan, Puerto Rico*, IEEE Catalog No. 99CH37011(C), 1999, 12a2 .

[6] Shur, M.S. Fjeldly, T. A. Shen, H., "AIM-Lab - A System for Conducting Semiconductor Device Characterization via the Internet", late news paper at 1999 International Conference on Microelectronic Test Structures (ICMTS 1999), Gothenburg, Sweden .

[7] Fjeldly, T.A. Shur, M.S. Shen, H. Ytterdal, T., "AIM-Lab: A System for -Remote Characterization of Electronic Devices and Circuits over the Internet", *Proc. 3rd IEEE Int. Caracas Conf. on Devices, Circuits and Systems (ICCDCS-2000), Cancun, Mexico*, IEEE Catalog No. 00TH8474C, 2000, pp. I43.1 -I43.6

[8] Smith, K. Strandman, J.O. Berntzen, R. Fjeldly, T.A. Shur, M.S., "Advanced Internet Technology in Laboratory Modules for Distance-Learning," *accepted for presentation at the American Society for Engineering Education Annual Conference & Exposition 2001*, ASEE'01.

[9] Fjeldly, T.A. Berntzen, R. Strandman, J.O. Shur, M.S. Jeppson, K. "LAB-on-WEB" *http://www.lab-on-web.com/*

[10] Ytterdal, T. Wulff, C. Sæthre, T.A. Skjevlan, A., "Next Generation Lab" *http://ngl.fysel.ntnu.no*

[11] Lee, E.K.F. Gulak, P.G. "A CMOS field-programmable analog array ". *Solid-State Circuits, IEEE Journal of* , Volume: 26 Issue: 12 , Dec. 1991, pp: 1860 –1867

[12] Gulak, P.G. "Field-Programmable Analog Arrays: Past, present and future perspectives". 1995. IEEE.

[13] Klein, H.W. "Introductory EPAC: an Analog FPGA", *IMP Inc*. ISBN# 0-7803-2636-9

[14] Klein, H.W. "The EPAC Architecture: An Expert Cell Approach to Field Programmable Analog Devices" *Lattice Semiconductor Corp. Analog Integrated Circuits and Signal Processing 17*, 1998, pp 91-103.

[15] Wulff, C, Ytterdal, T "Programmable analog integrated circuit for us in remotely operated laboratories". International Conference on Engineering Education (ICEE-2002). Found in Appendix VII and on the CDROM

[16] Europractice, http://www.europractice.com

[17] Wulff, C. "Mixed-Signal Design using Mentor Graphics", Fall 2002 Department of Physical Electronics, NTNU. Found on the CDROM

[18] Austria Micro Systems, http://www.austriamicrosystems.com

# Appendices

## Appendix I: Successive Approximation ADC

An ADC of this type converters the input value by performing a binary search. It starts by comparing the input value to the digital word in the middle of the range, for an 8-bit converter this is 128, if it is higher we know that the digital representation of the input value is between 128-255, if it is lower it is between 0-127. An 8-bit successive approximation analog to digital converter (ADC) is built up of one digital to analog converter (DAC), one comparator and a successive approximation register modeled in software. The software routine performs a binary search for the correct digital word. A flowchart for the successive approximation ADC architecture is provided in figure 1. A block diagram of this ADC is shown in figure 2.

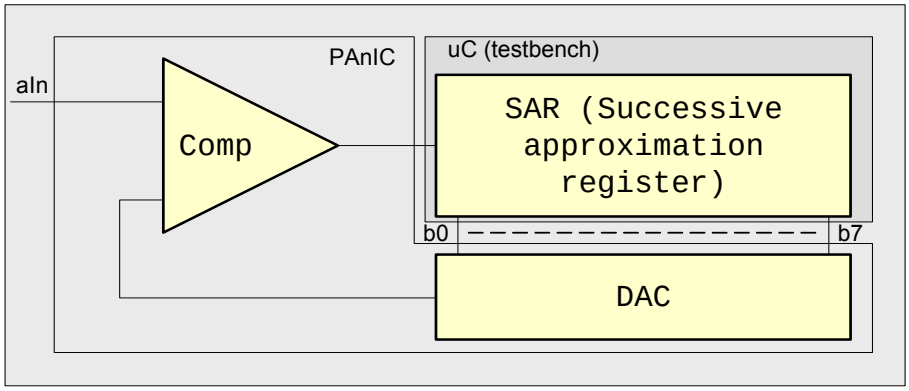



*Figure 1 ADC block diagram*

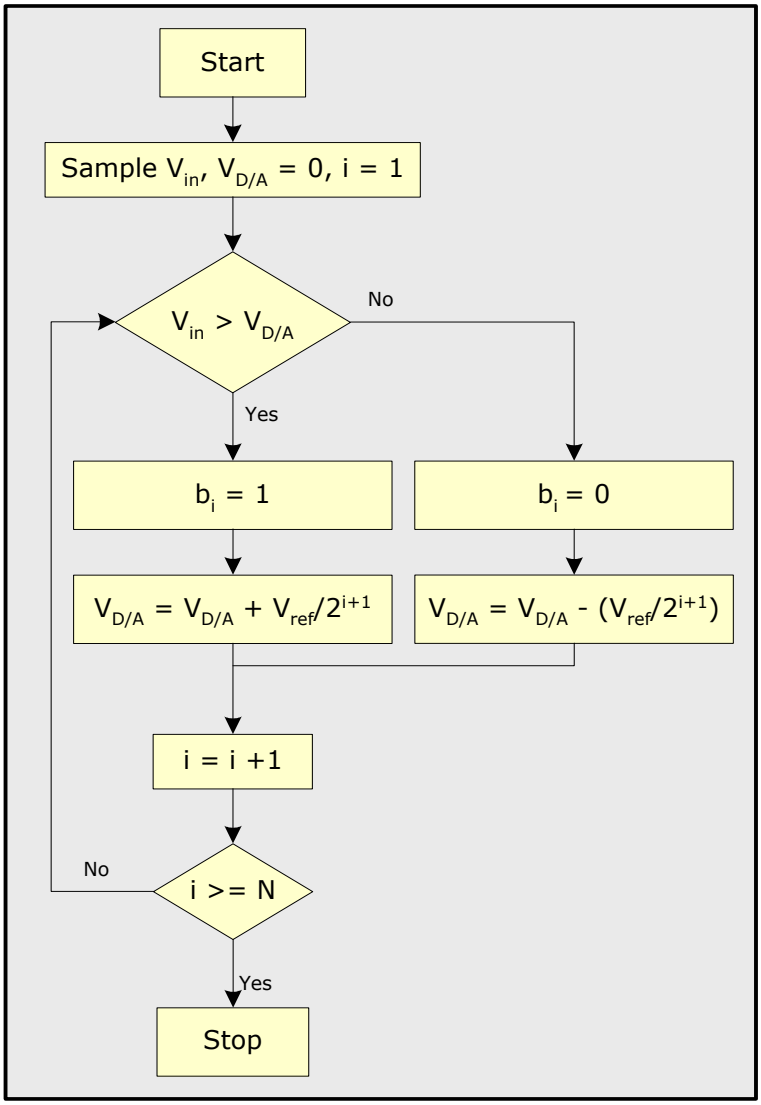



*Figure 2 Successive approximation ADC flowchart*

# Appendix II: Serial Peripheral Interface

SPI is a widely used interface, and is featured in many micro-controllers and IC's. The general idea behind a SPI is to have two shift registers connected together with the `miso` (master in slave out) and `mosi` (master out slave in) signals as show in the figure below. The master controls the transfer through the `sck` signal. When `sck` goes high, the master writes its MSB to the `mosi` signal and shifts one position, the slave writes its MSB to the `miso` signal and shifts one position. Both master and slave write to their LSB from their respective input signals. This way the data is shifted from master to slave and from slave to master, thus providing bi-directional data transfer.

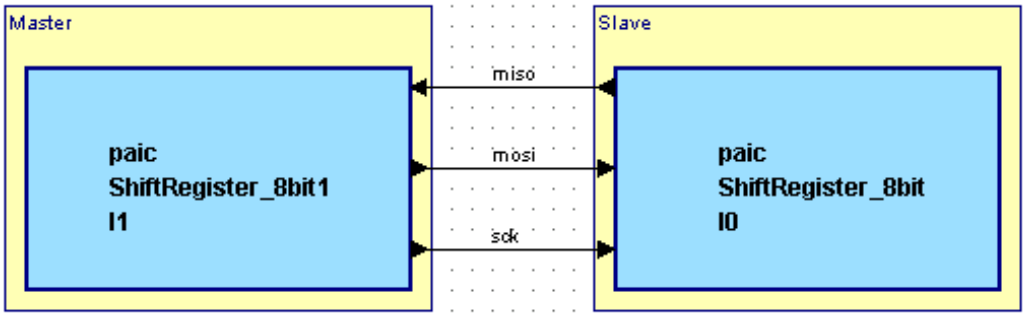
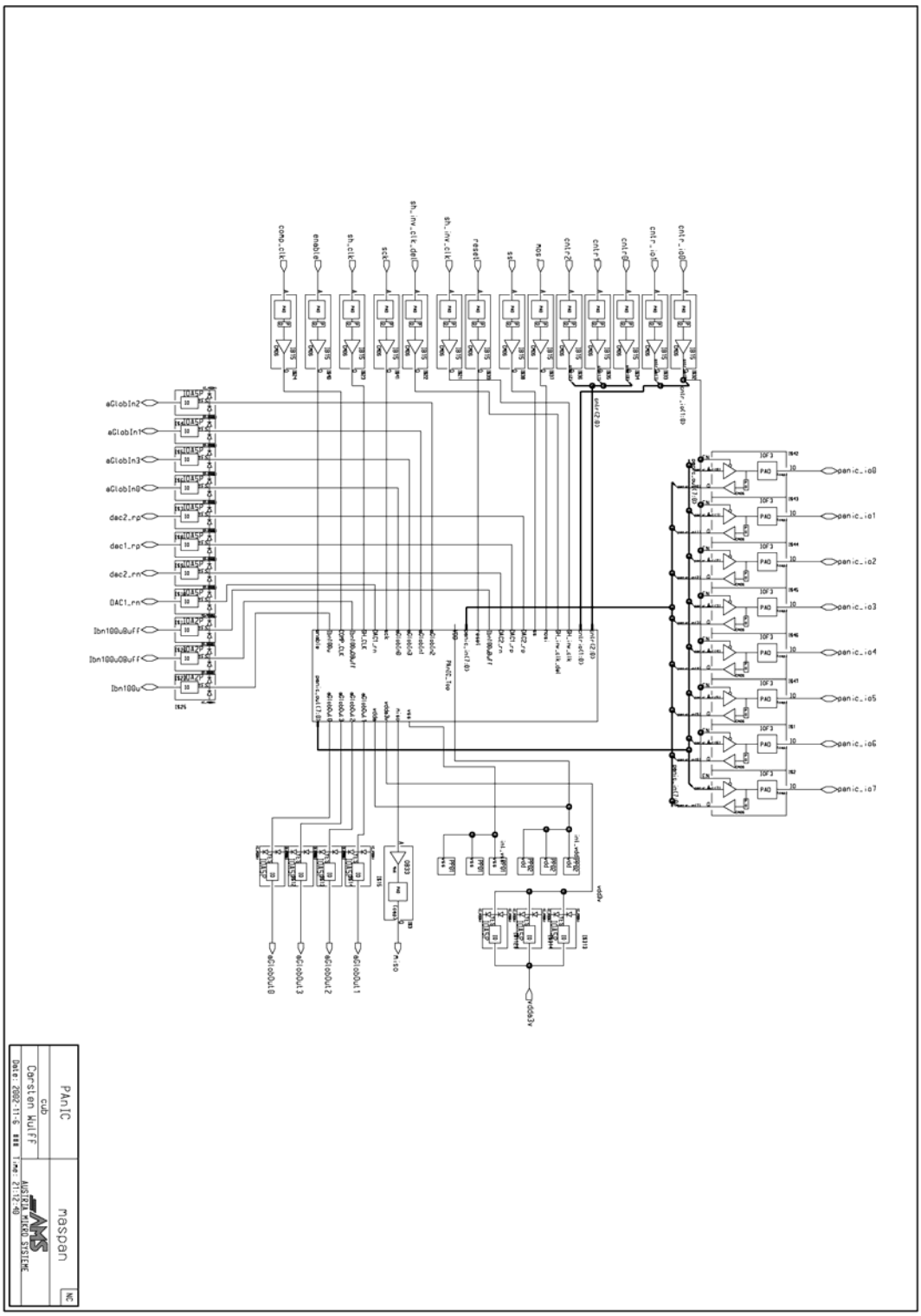
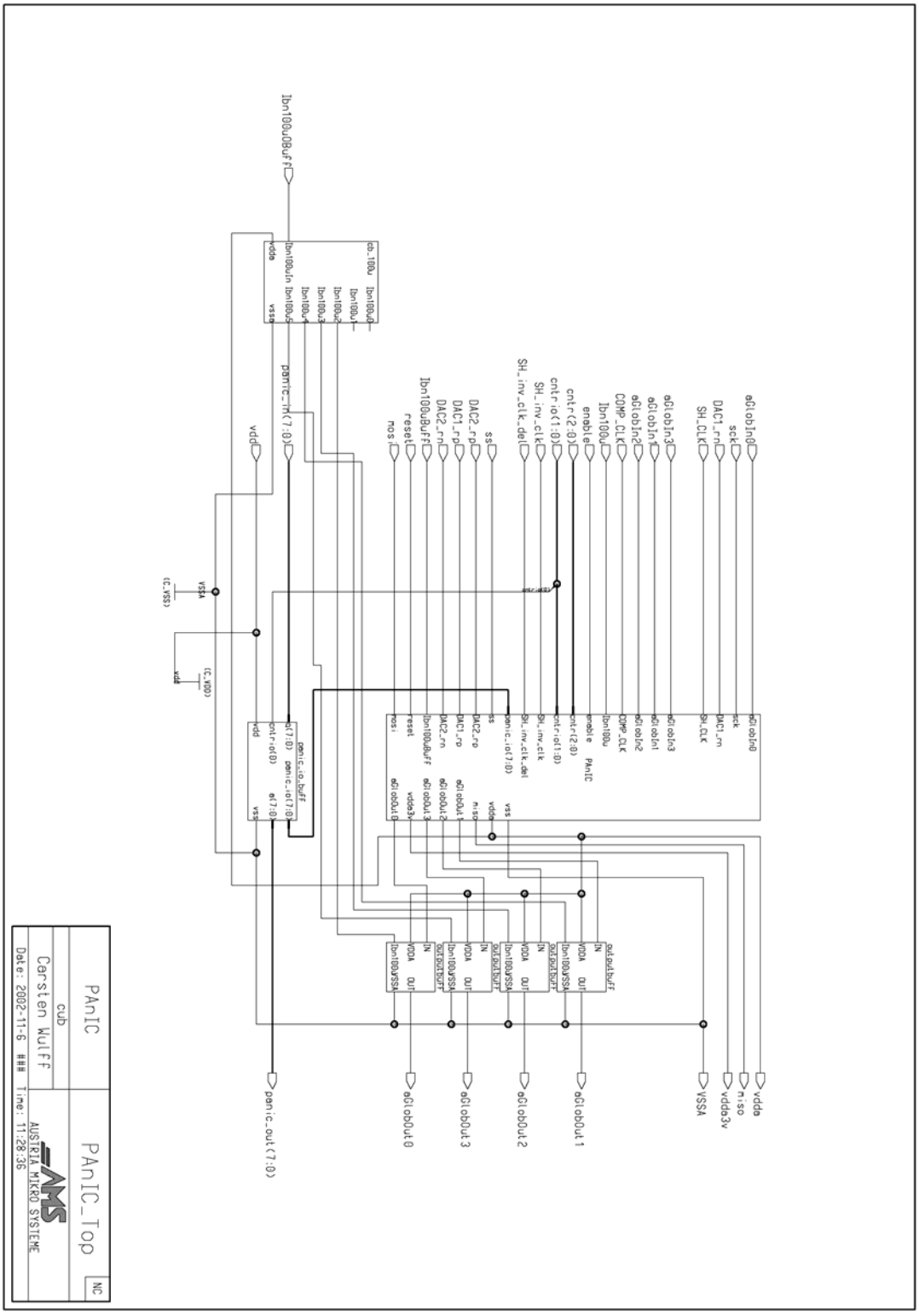



*Figure 1 SPI block diagram*

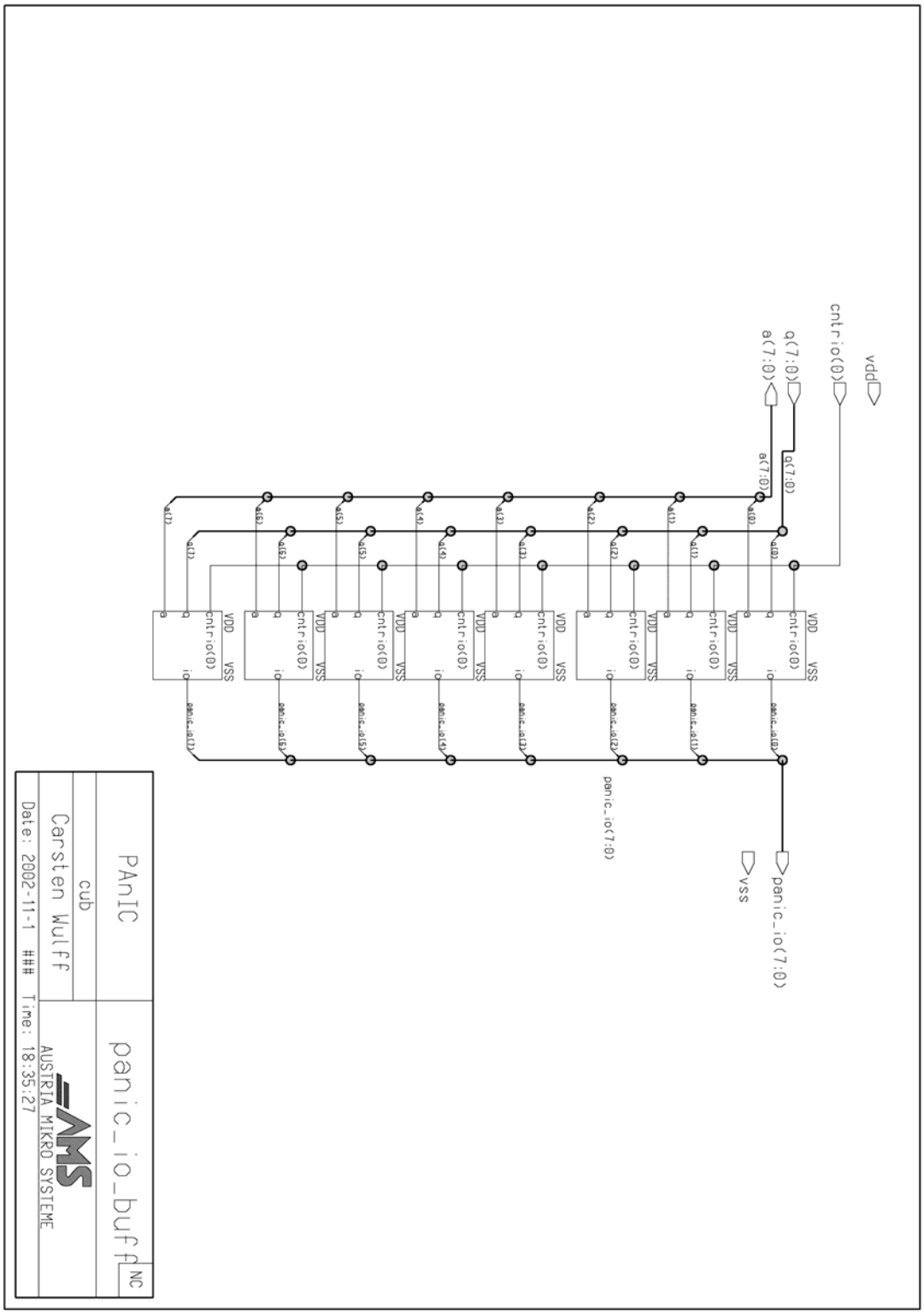
# Appendix III: Schematics



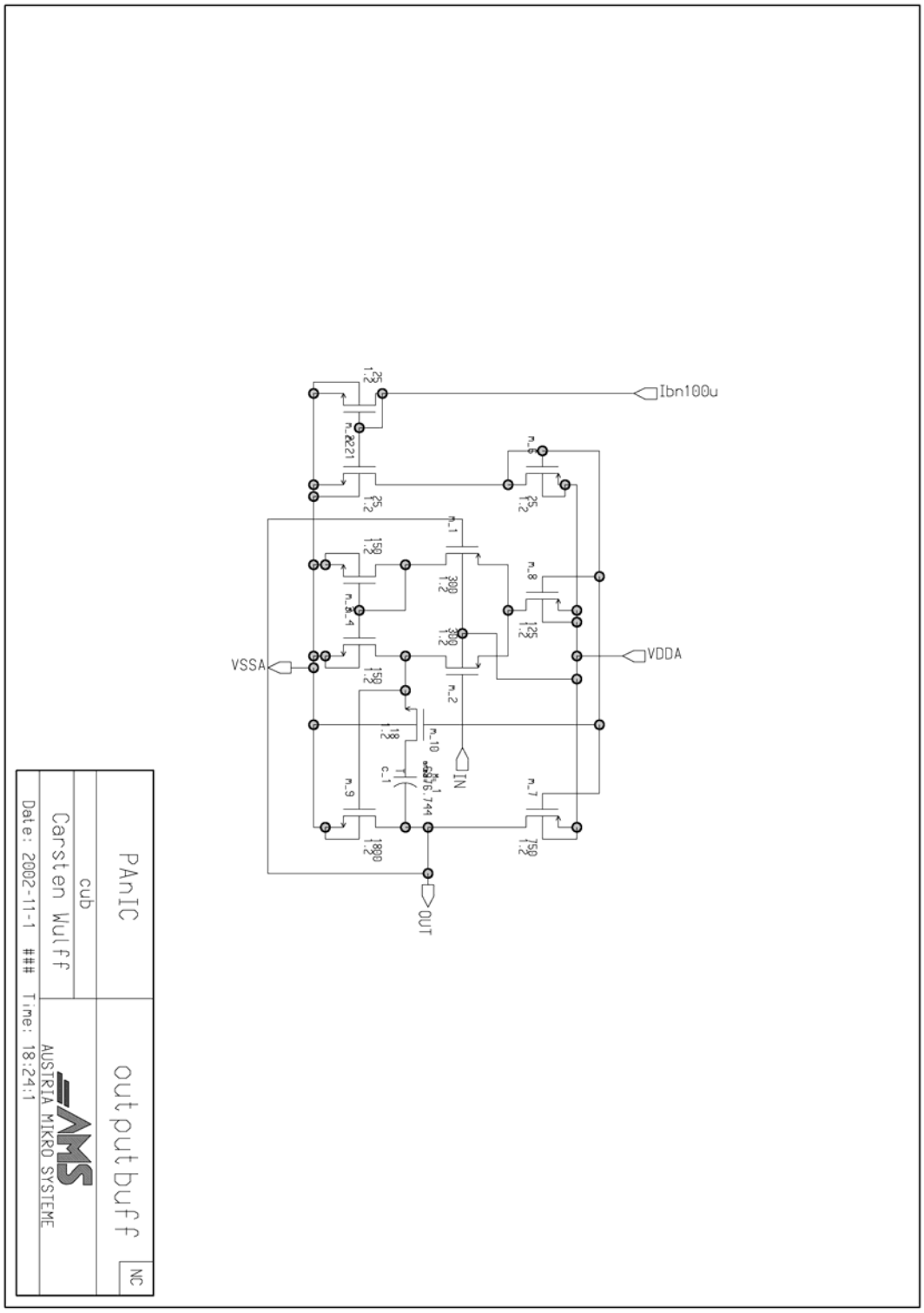

*Schematic 1 maspan*



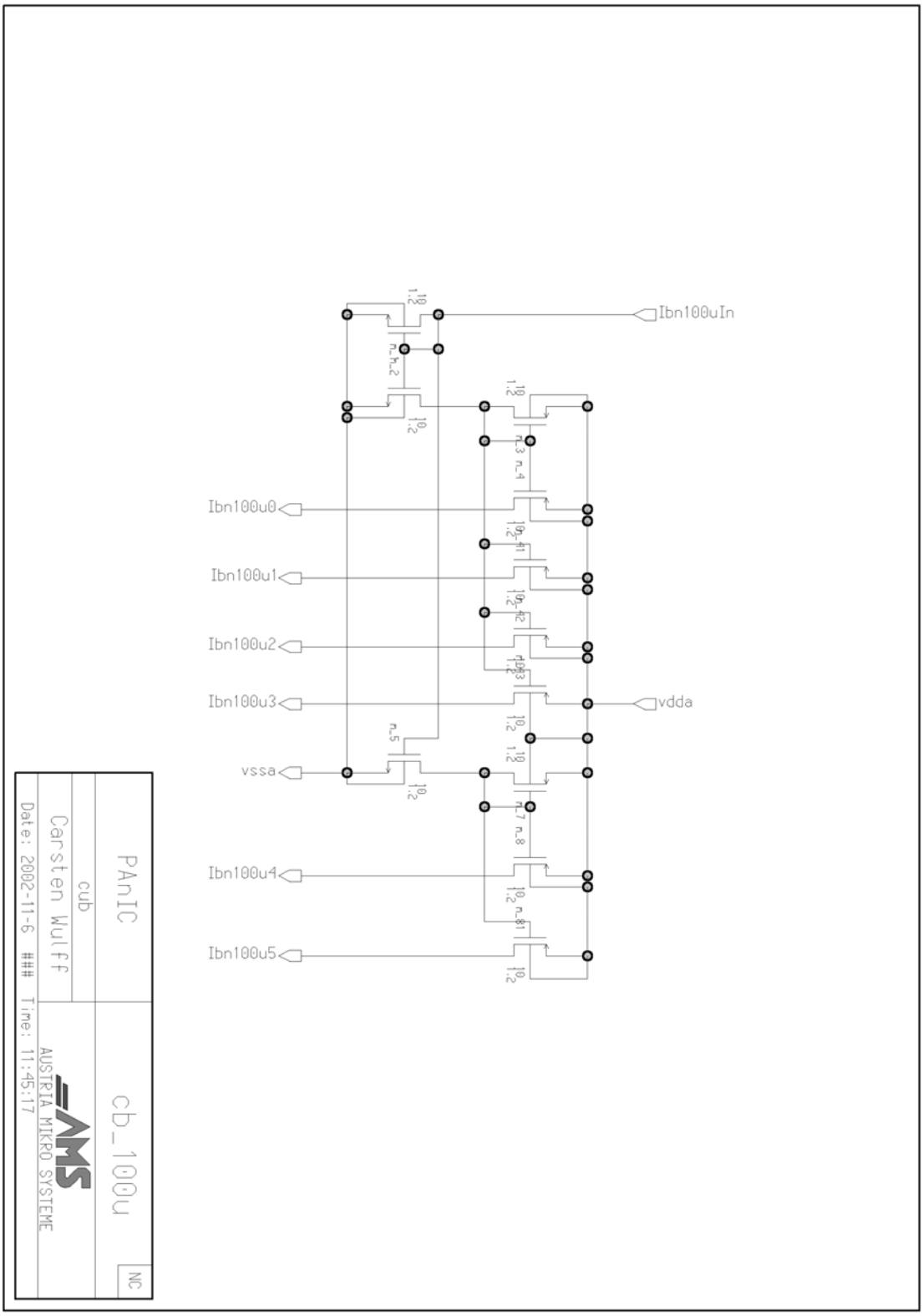

*Schematic 2 panic_top*



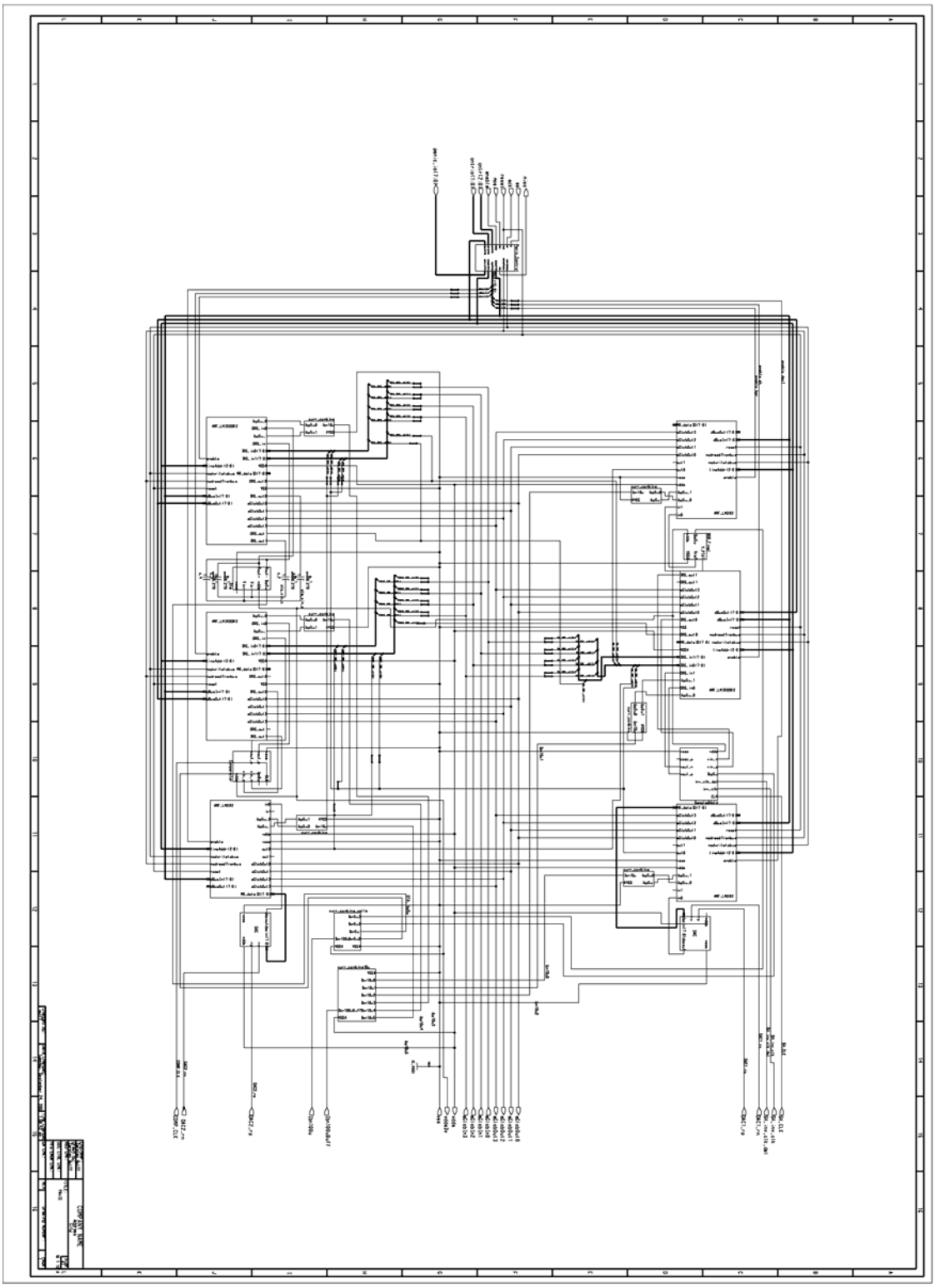

*Schematic 3 panic_io_buff*



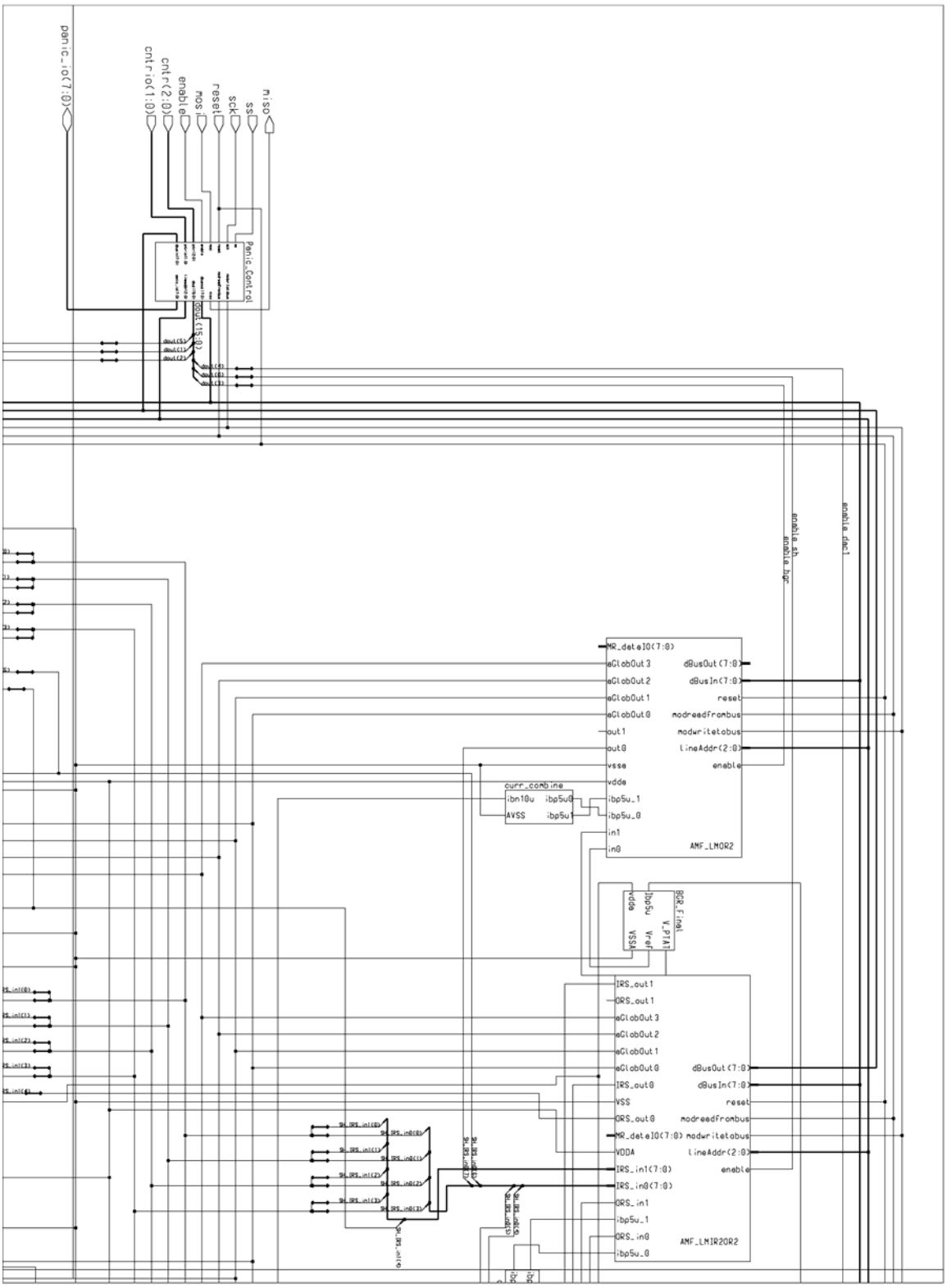

*Schematic 4 outputbuff*



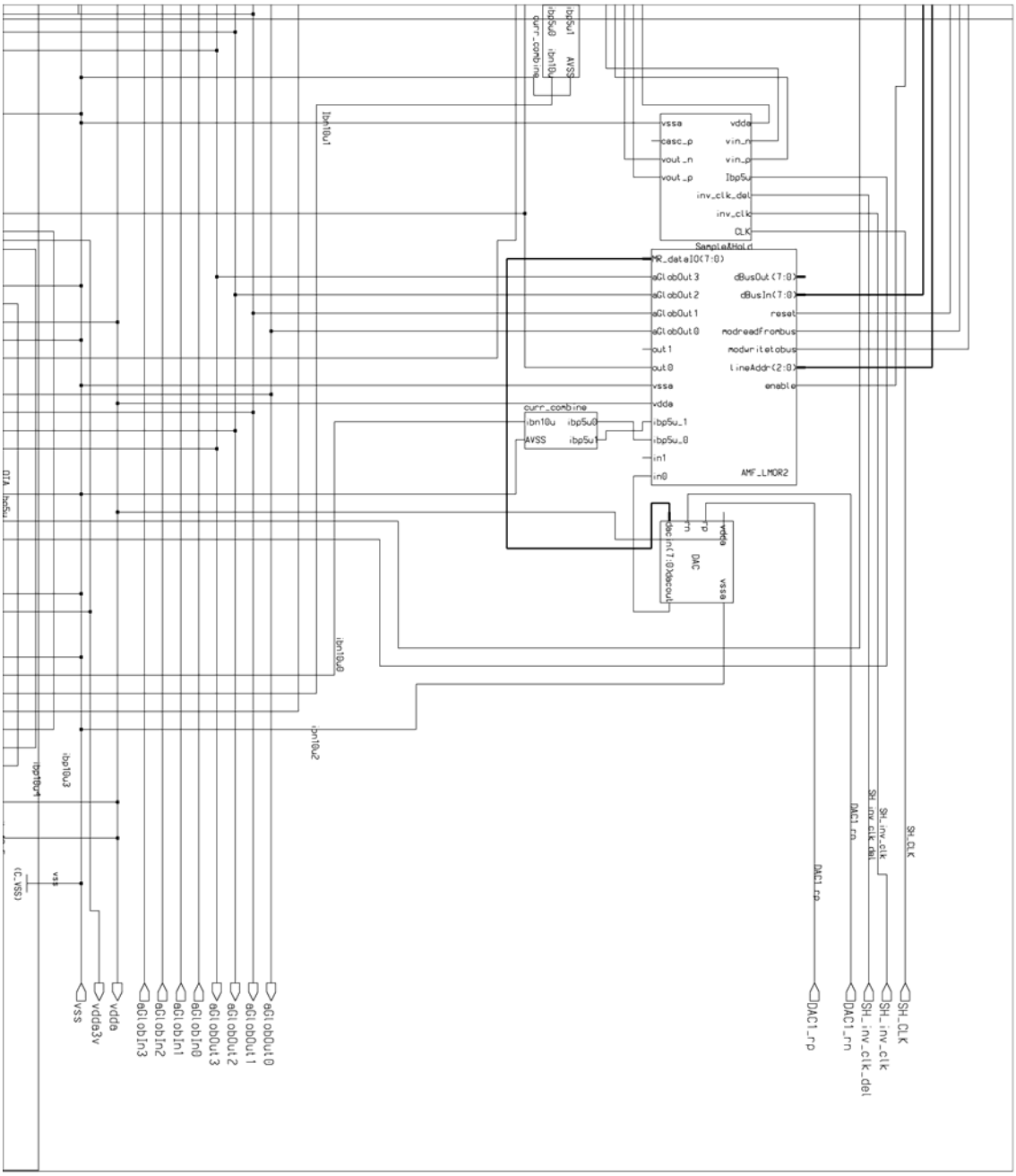

*Schematic 5 cb_100u*



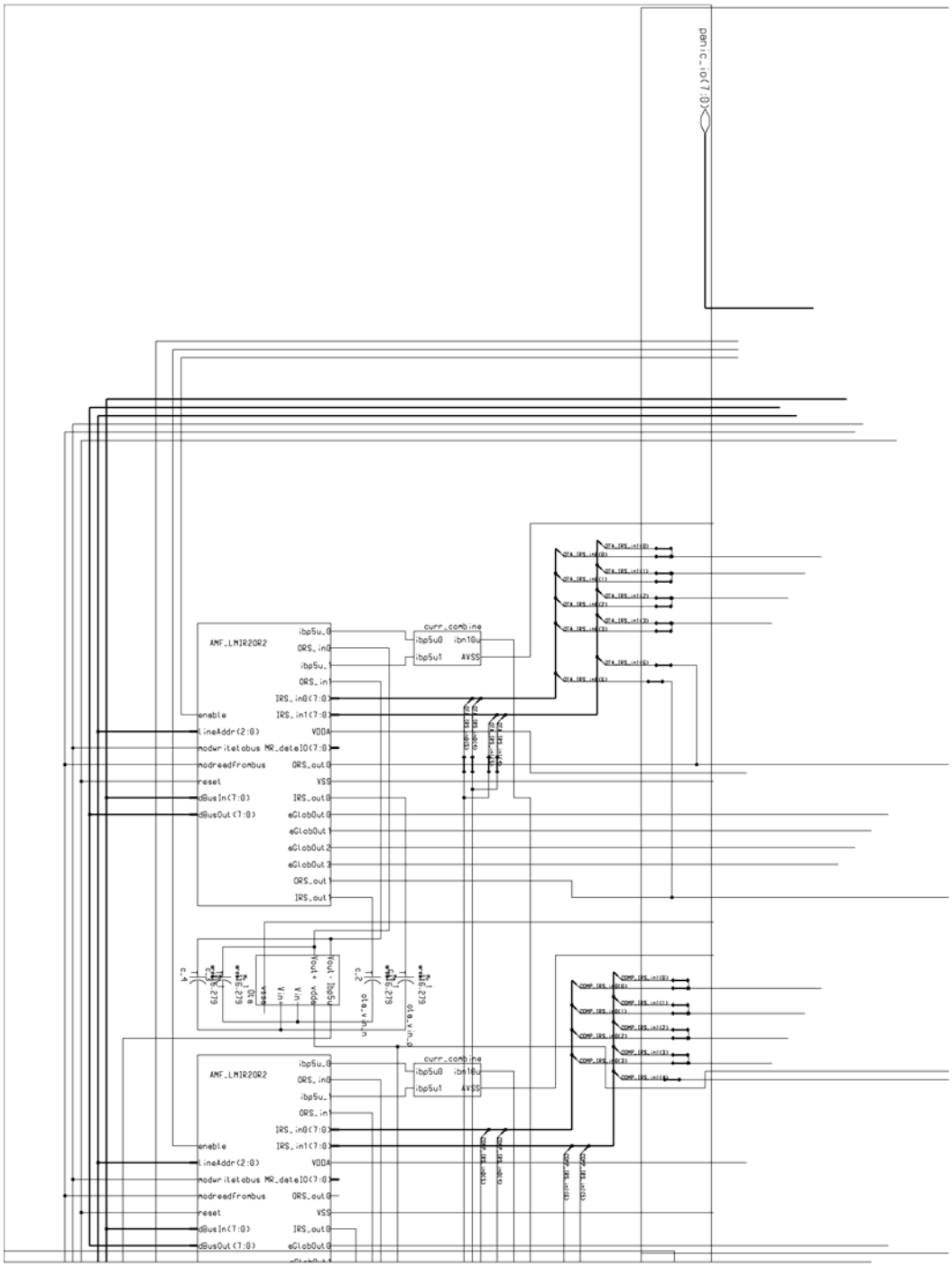
*Schematic 6 panic*



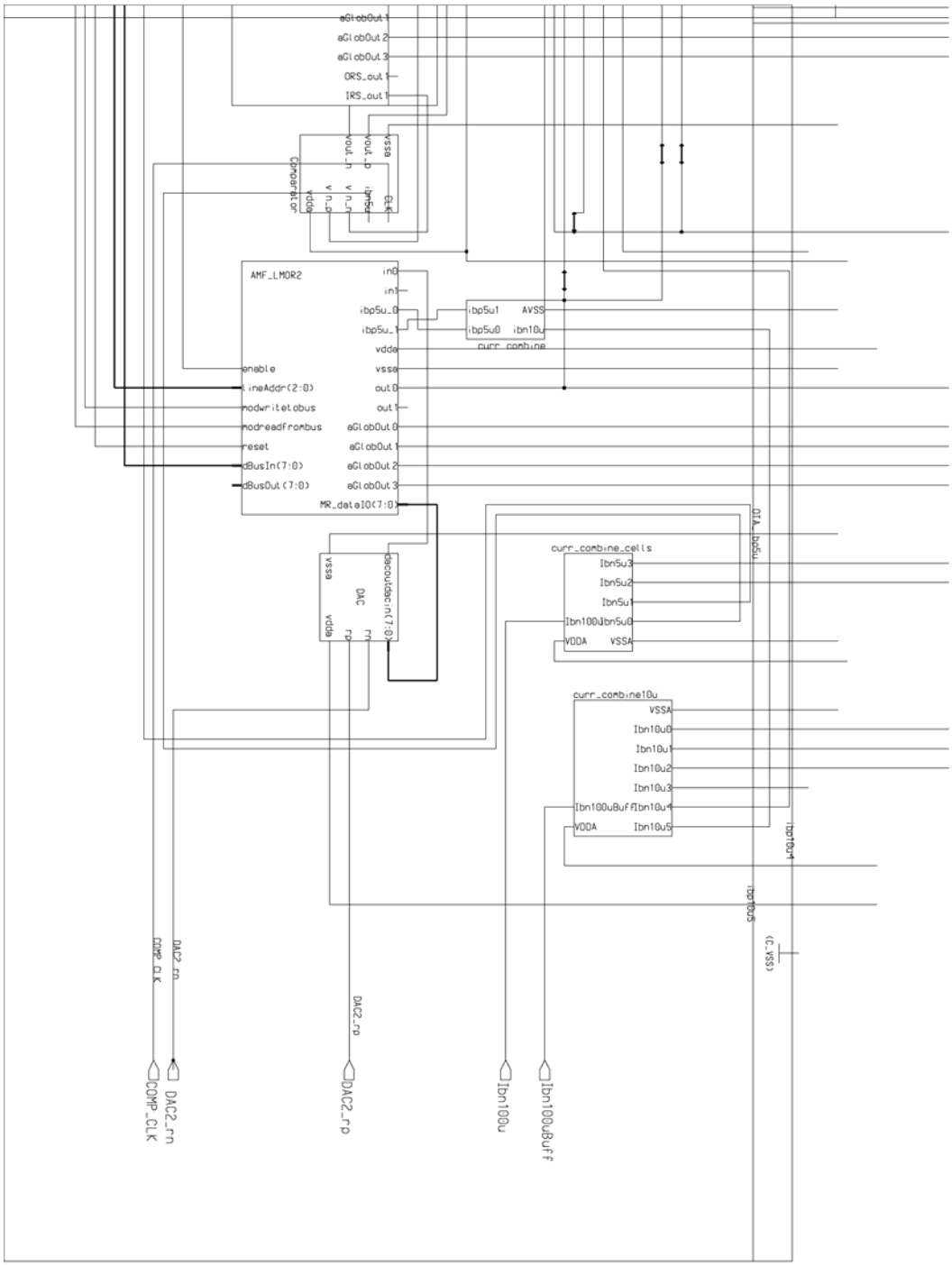

*Schematic 7 panic (1/4) top left*



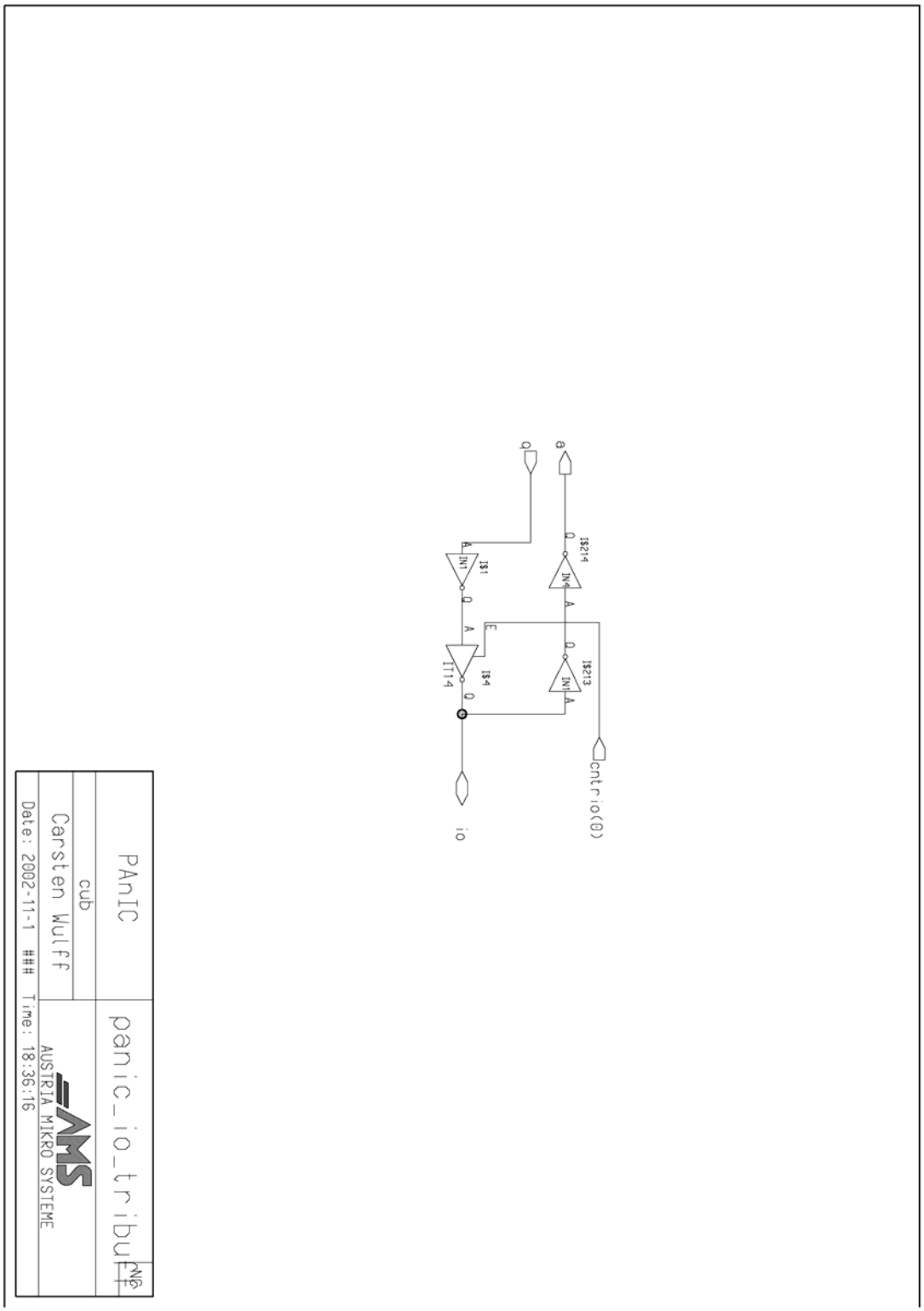

*Schematic 8 panic (2/4) top right*



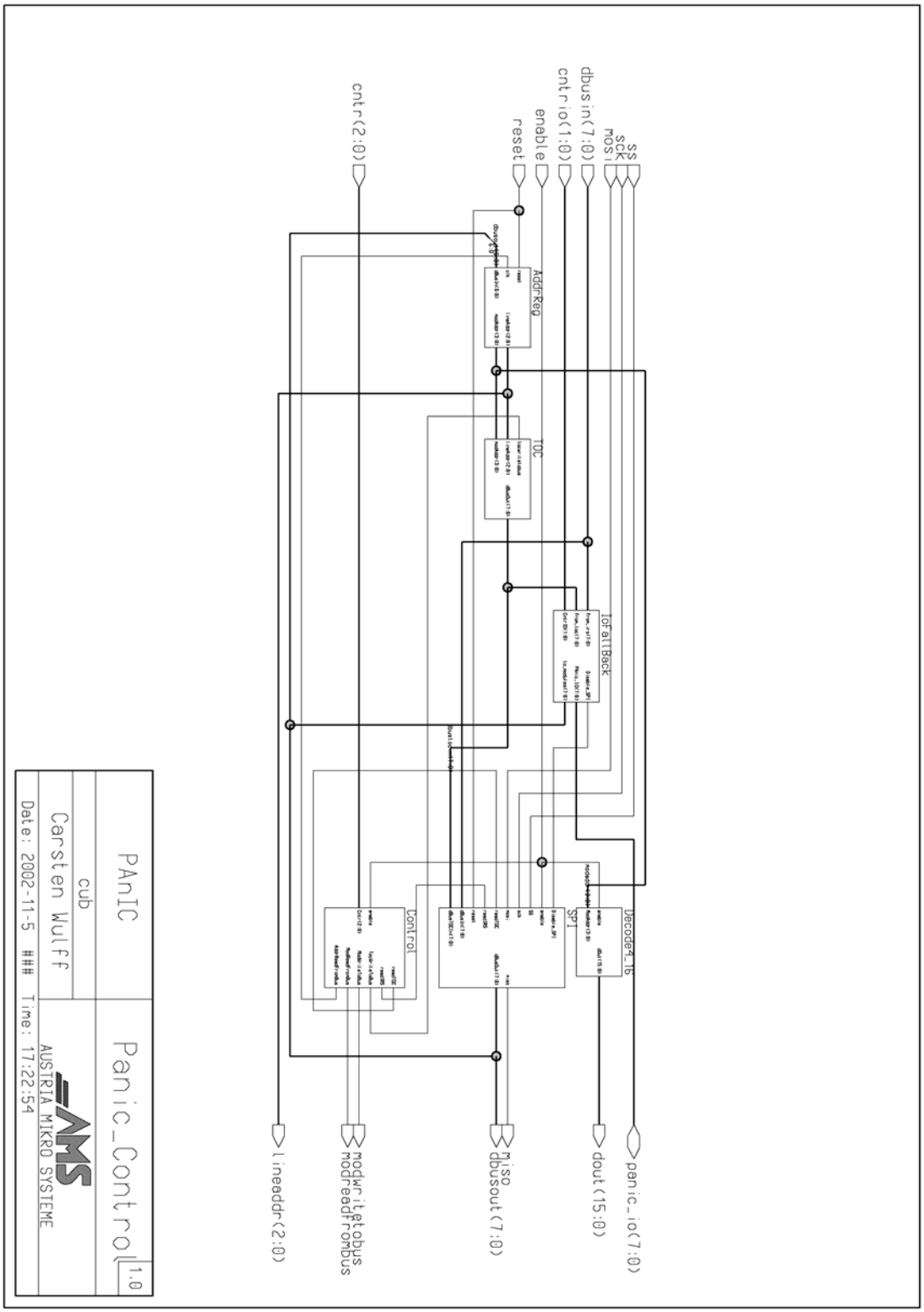

*Schematic 9 panic (3/4) bottom left*



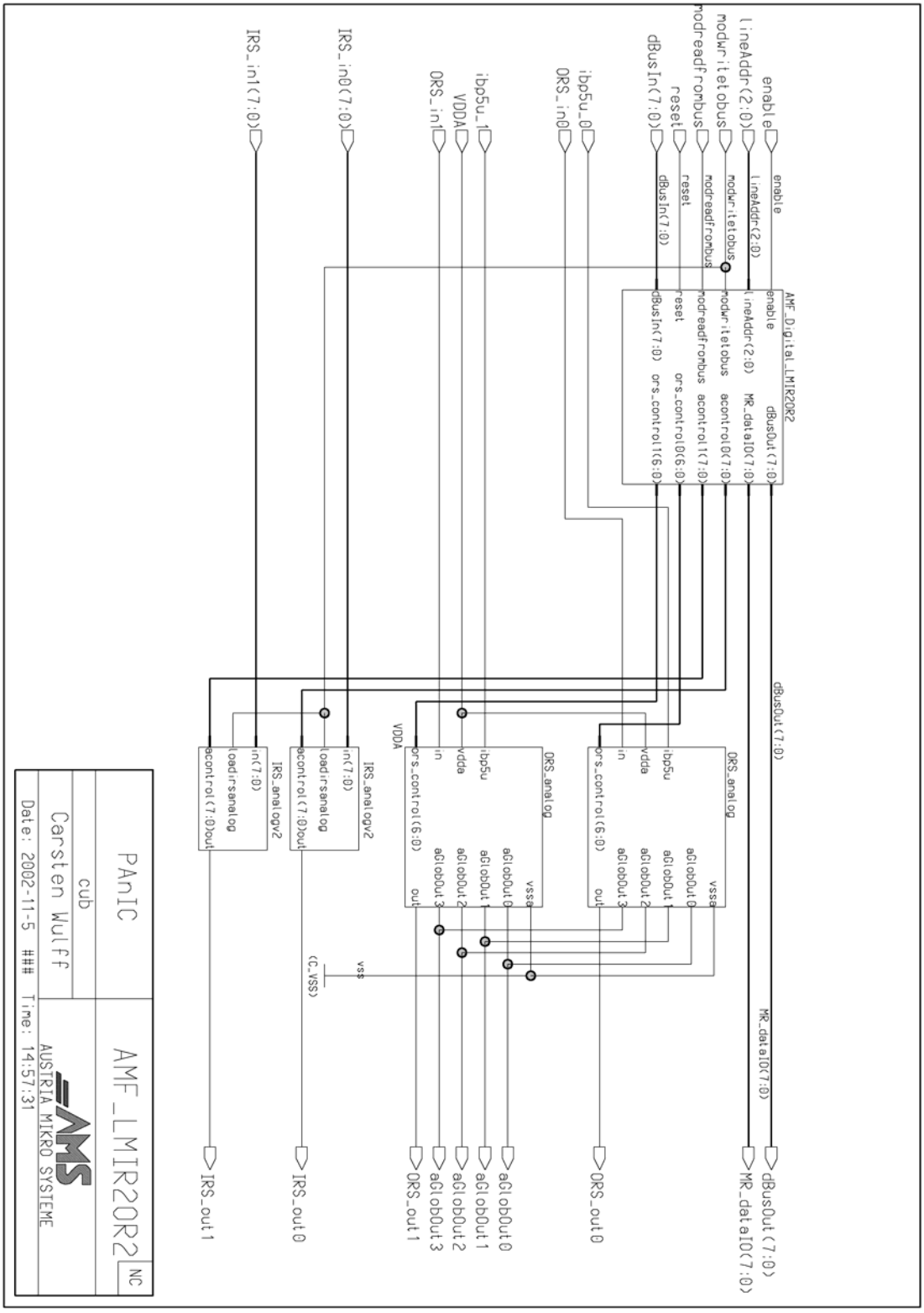

*Schematic 10 panic (4/4) bottom right*



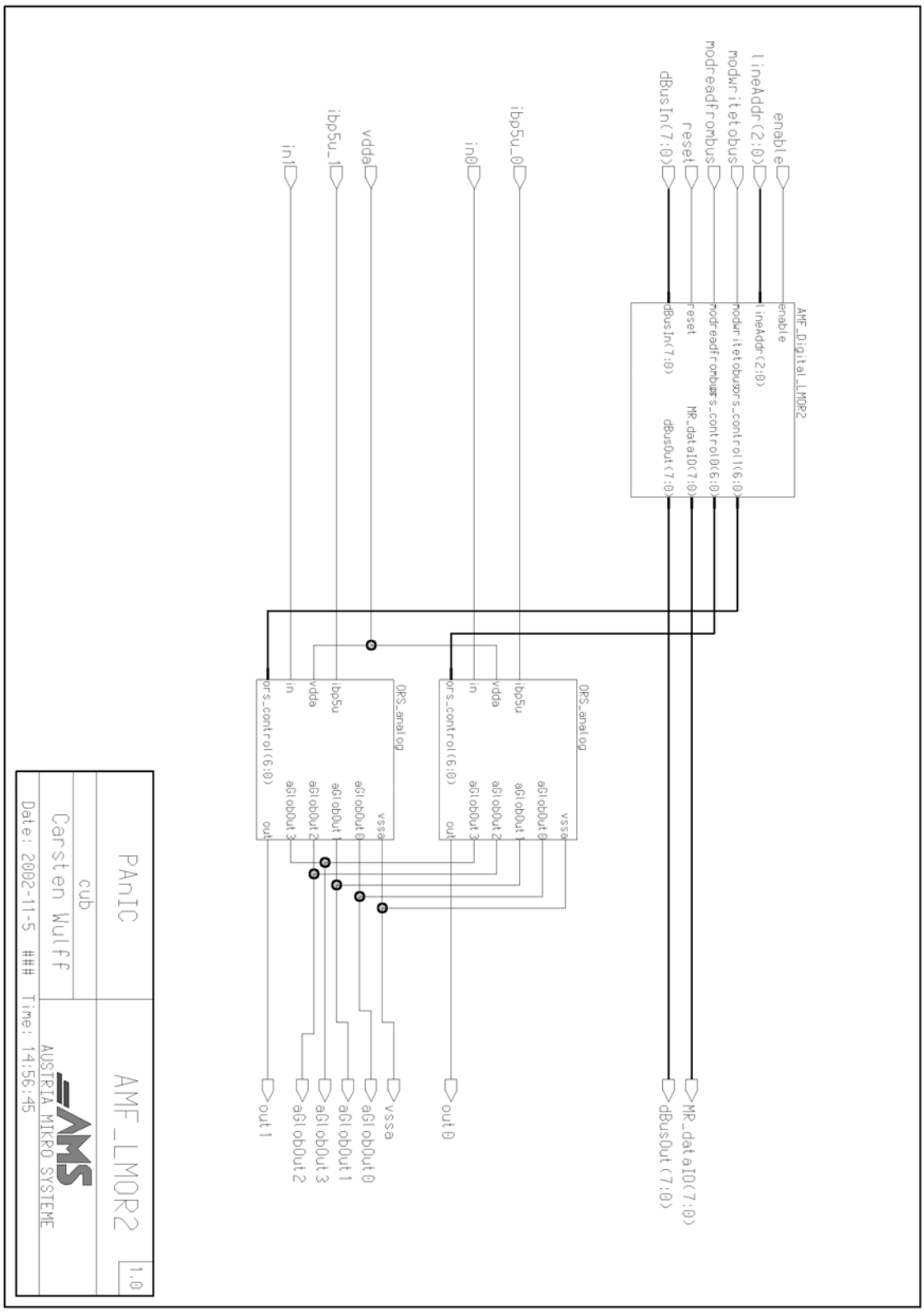

*Schematic 11 panic_io_tribuff*



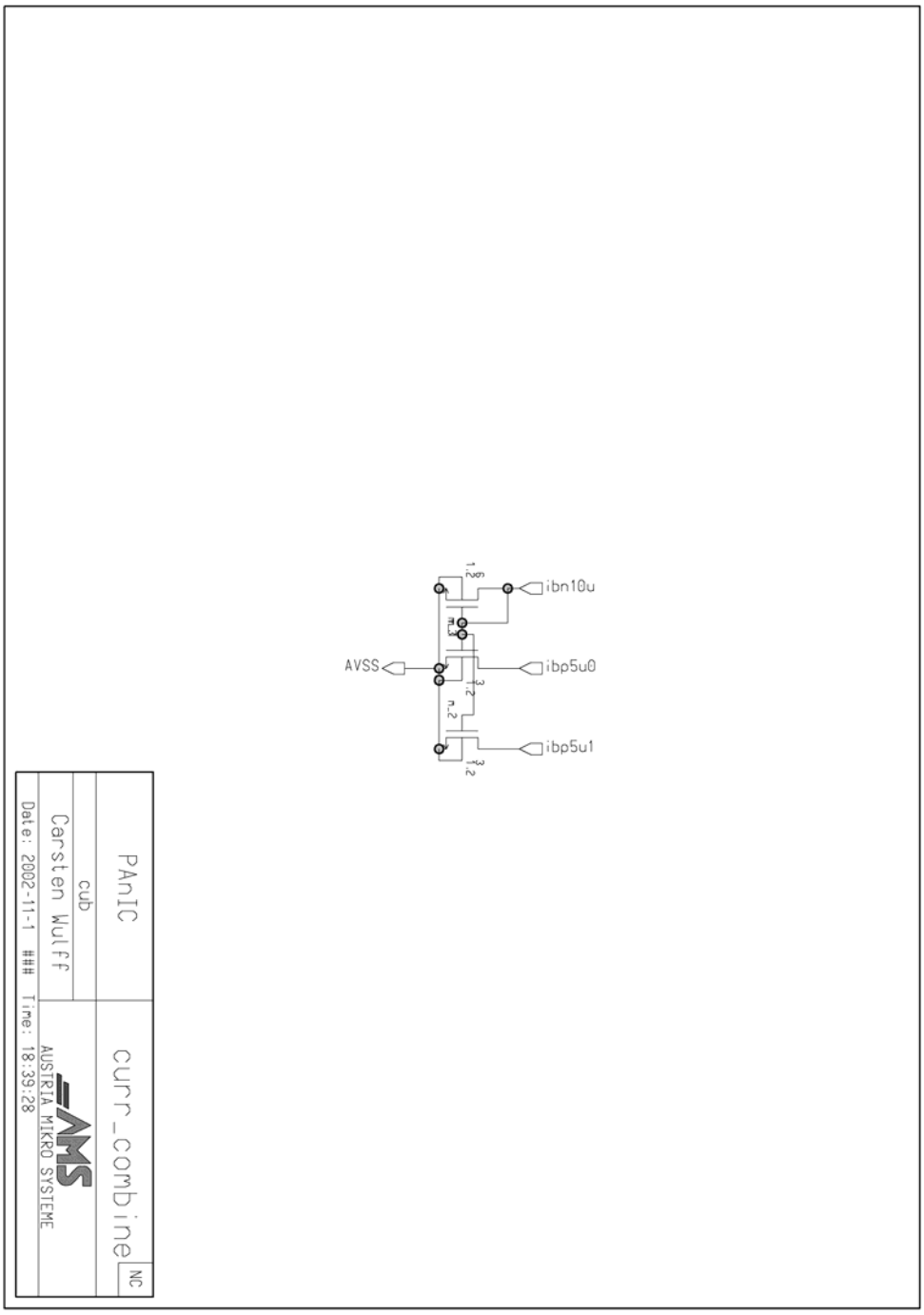

*Schematic 12 Panic_Control*



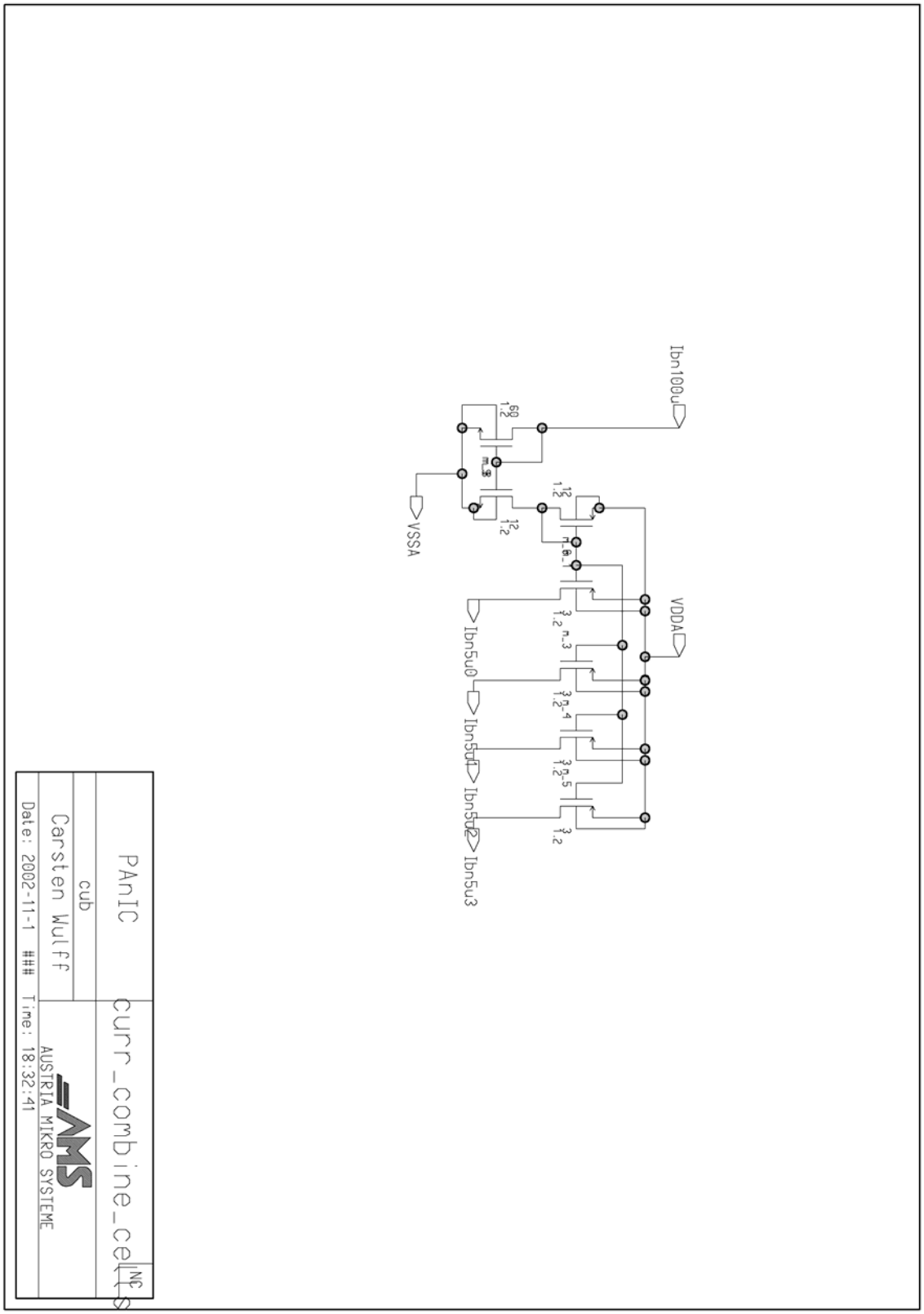

*Schematic 13 AMF_LMIR2OR2*



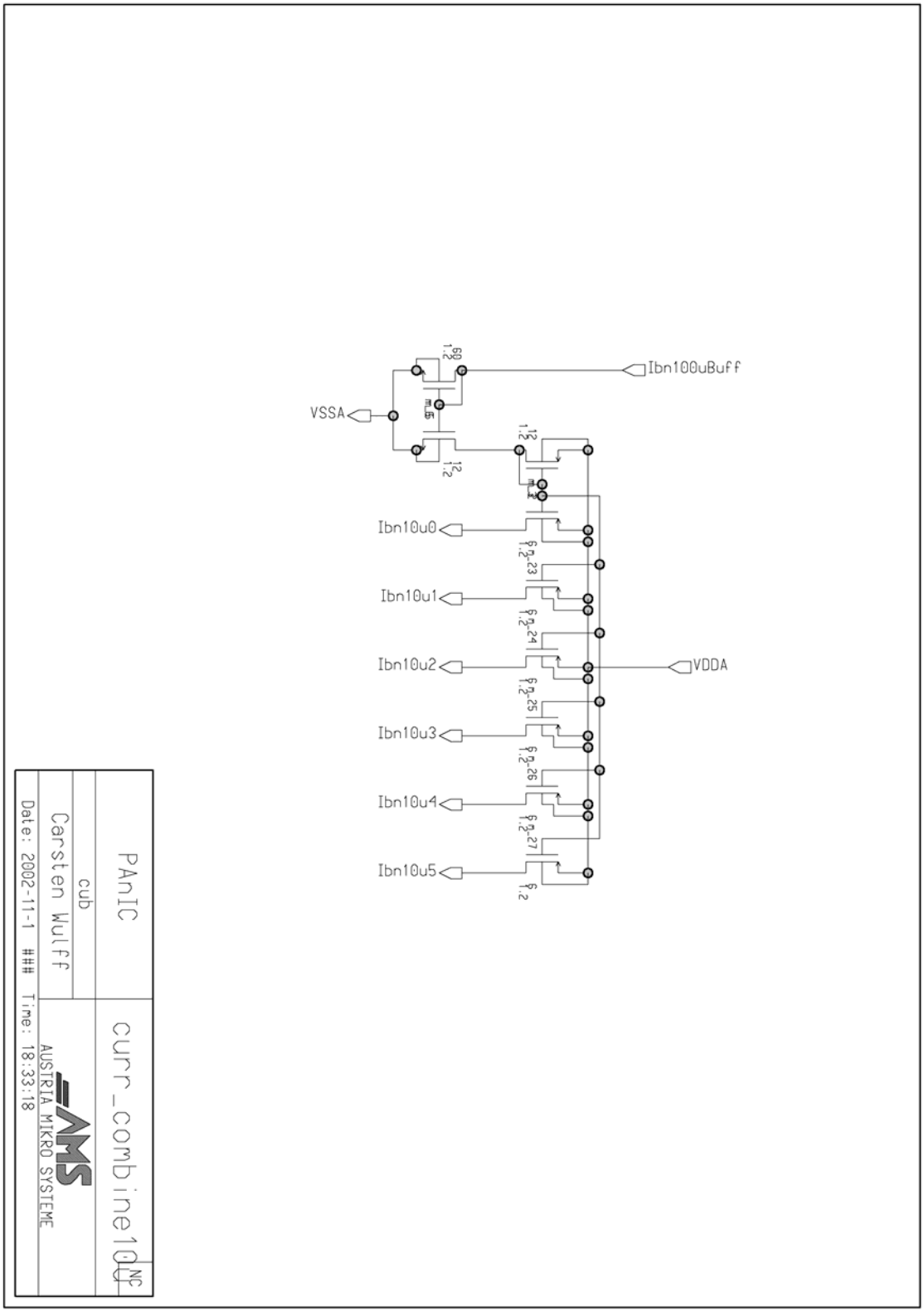

*Schematic 14 AMF_LMOR2*



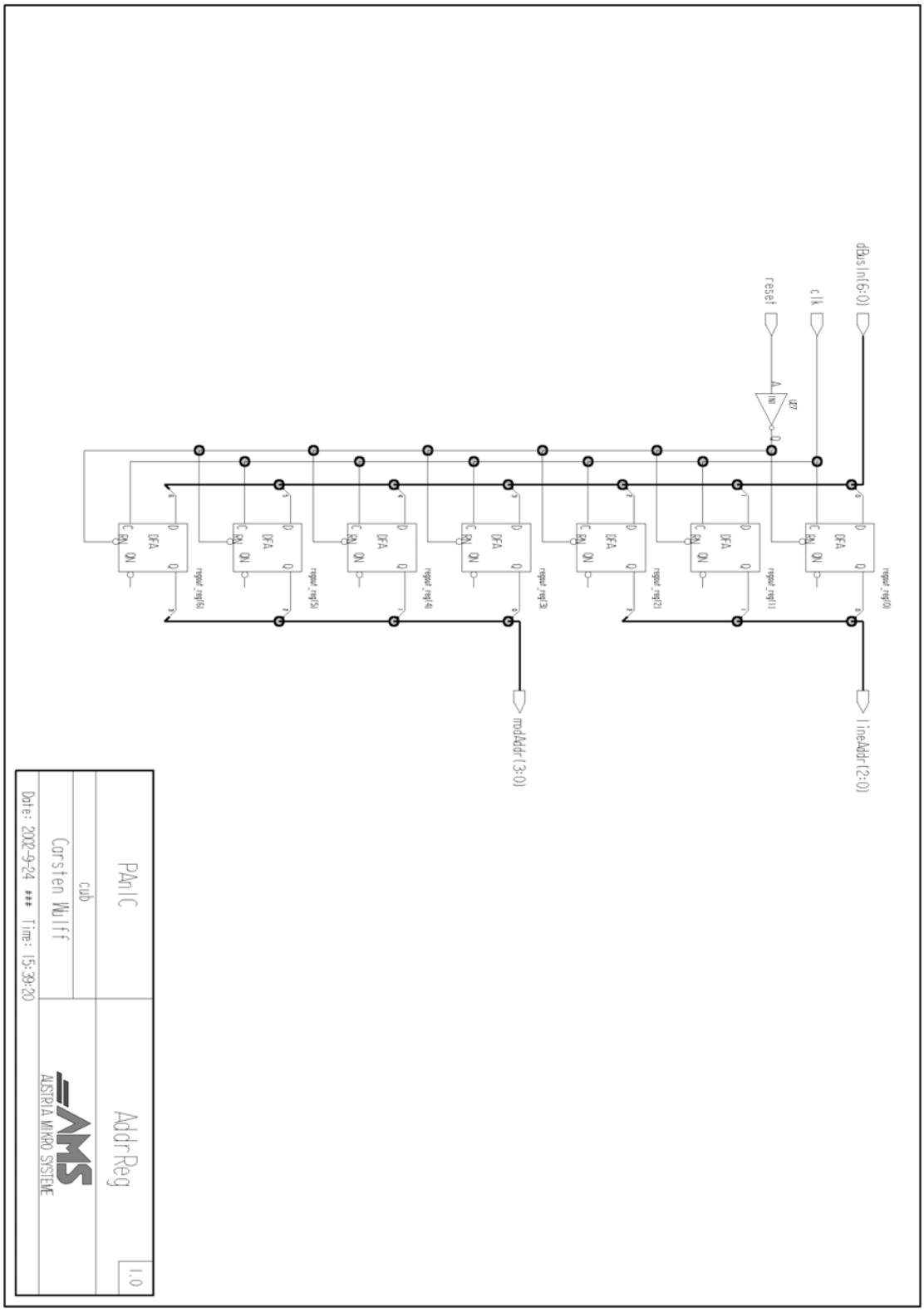

*Schematic 15 curr_combine*



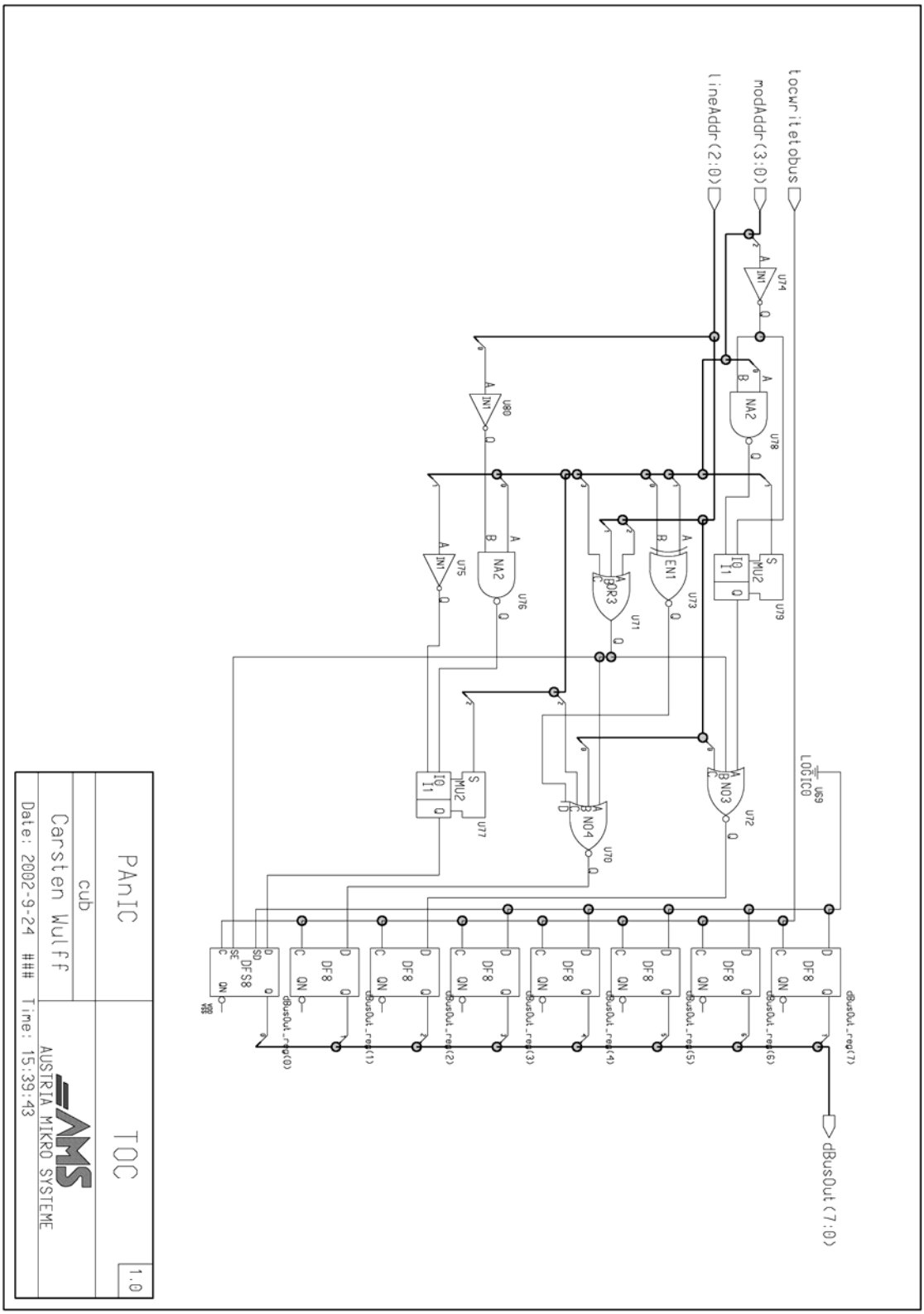

*Schematic 16 curr_combine_cells*



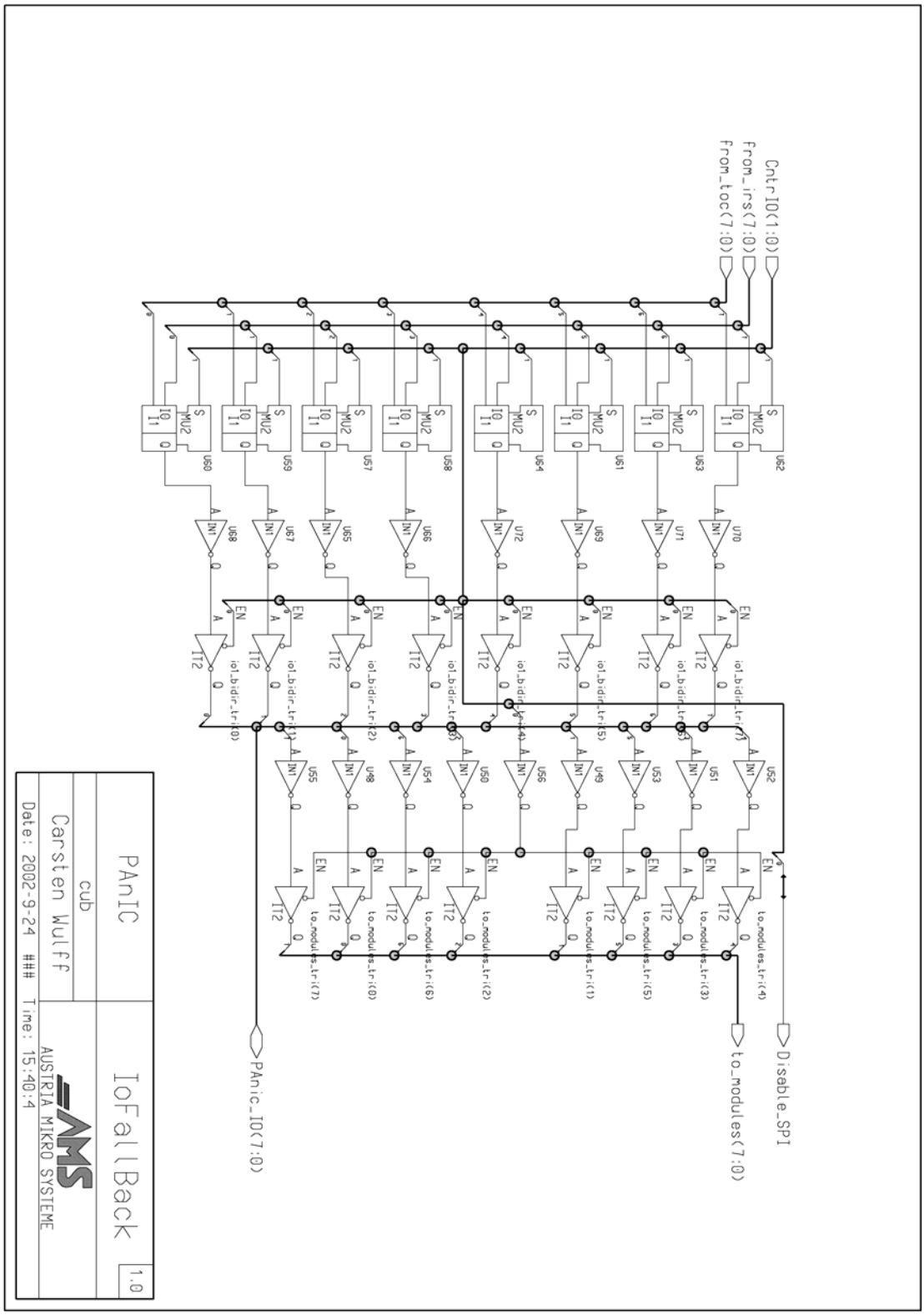

*Schematic 17 curr_combine10u*



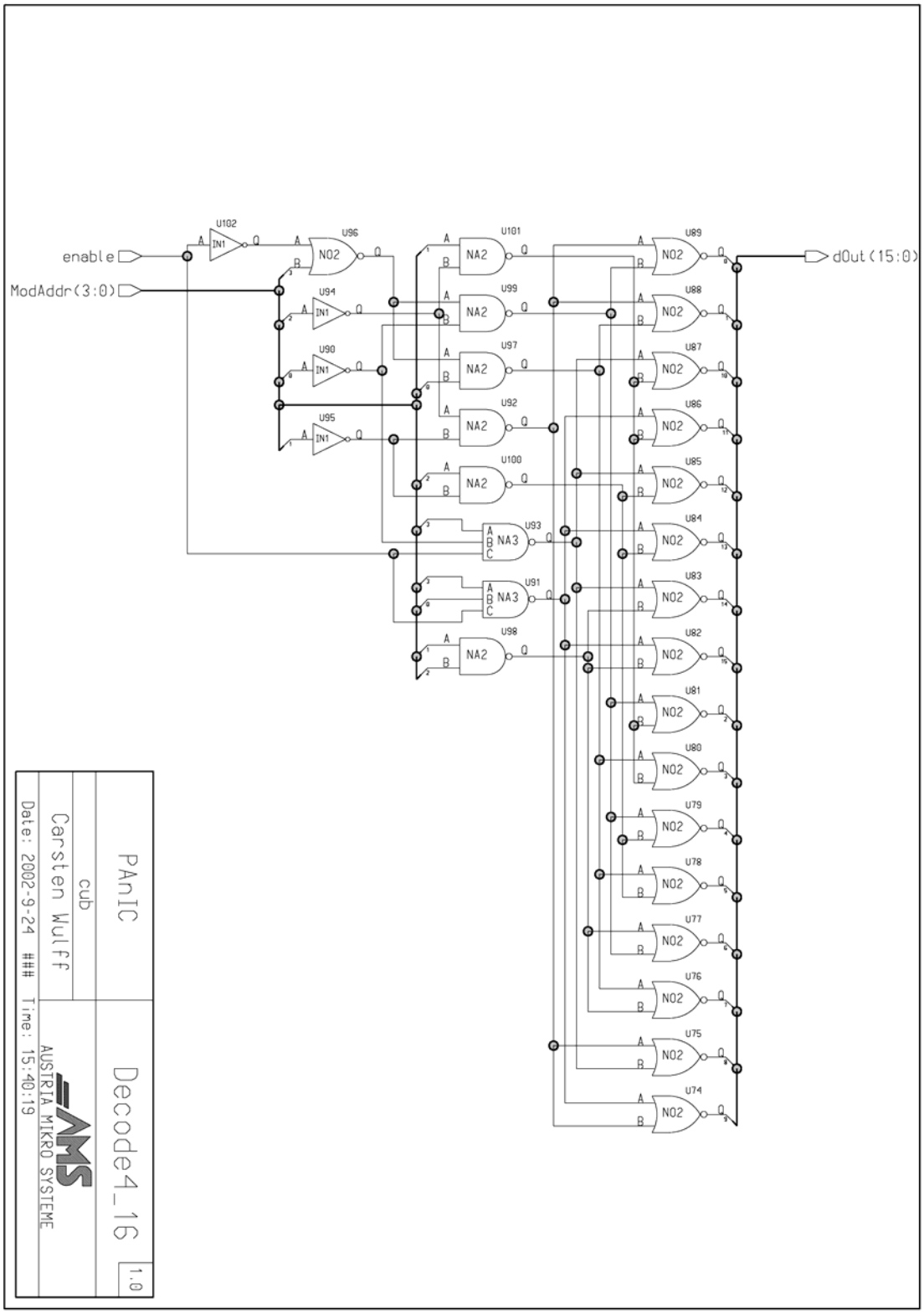

*Schematic 18 AddrReg*



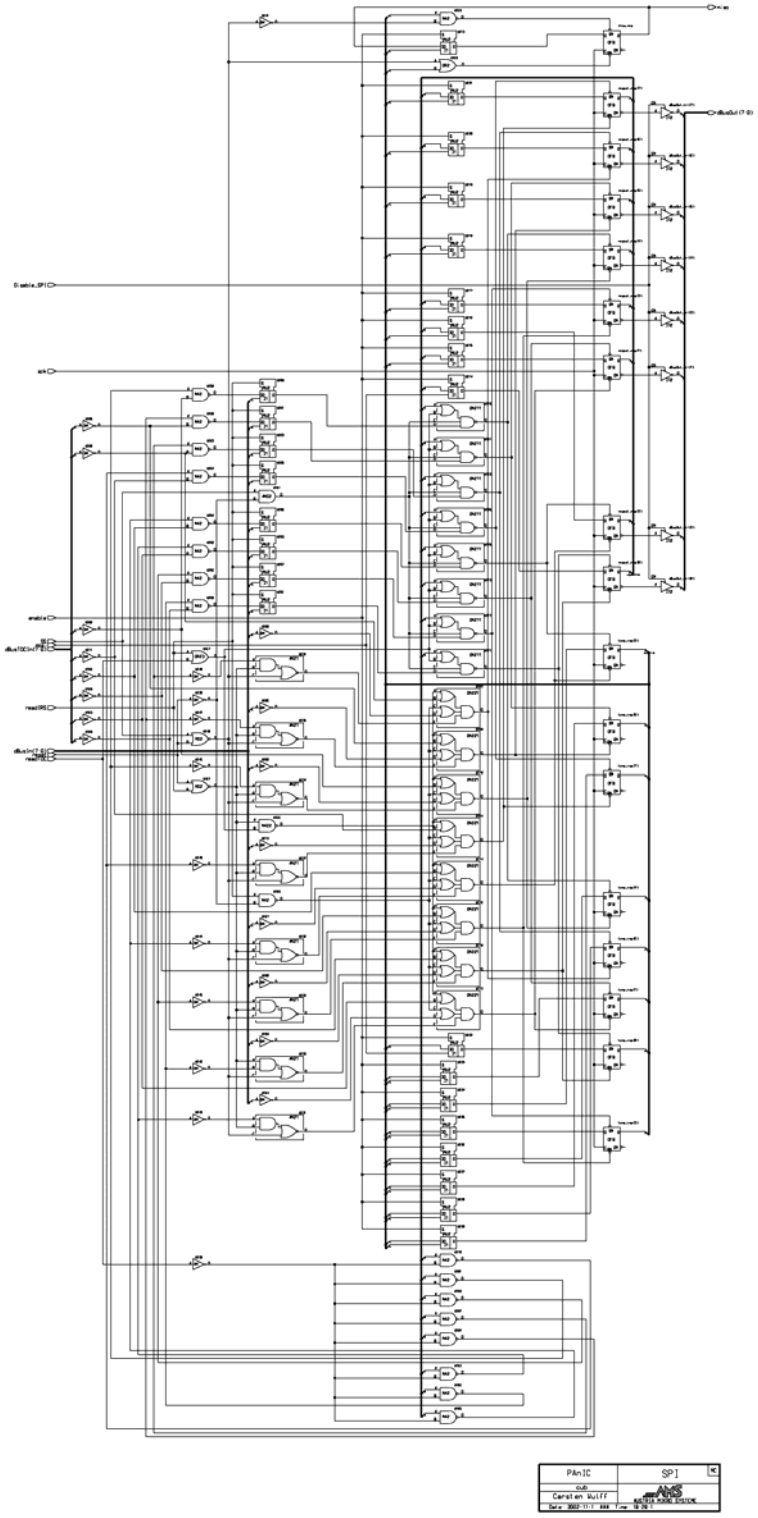

*Schematic 19 TOC*



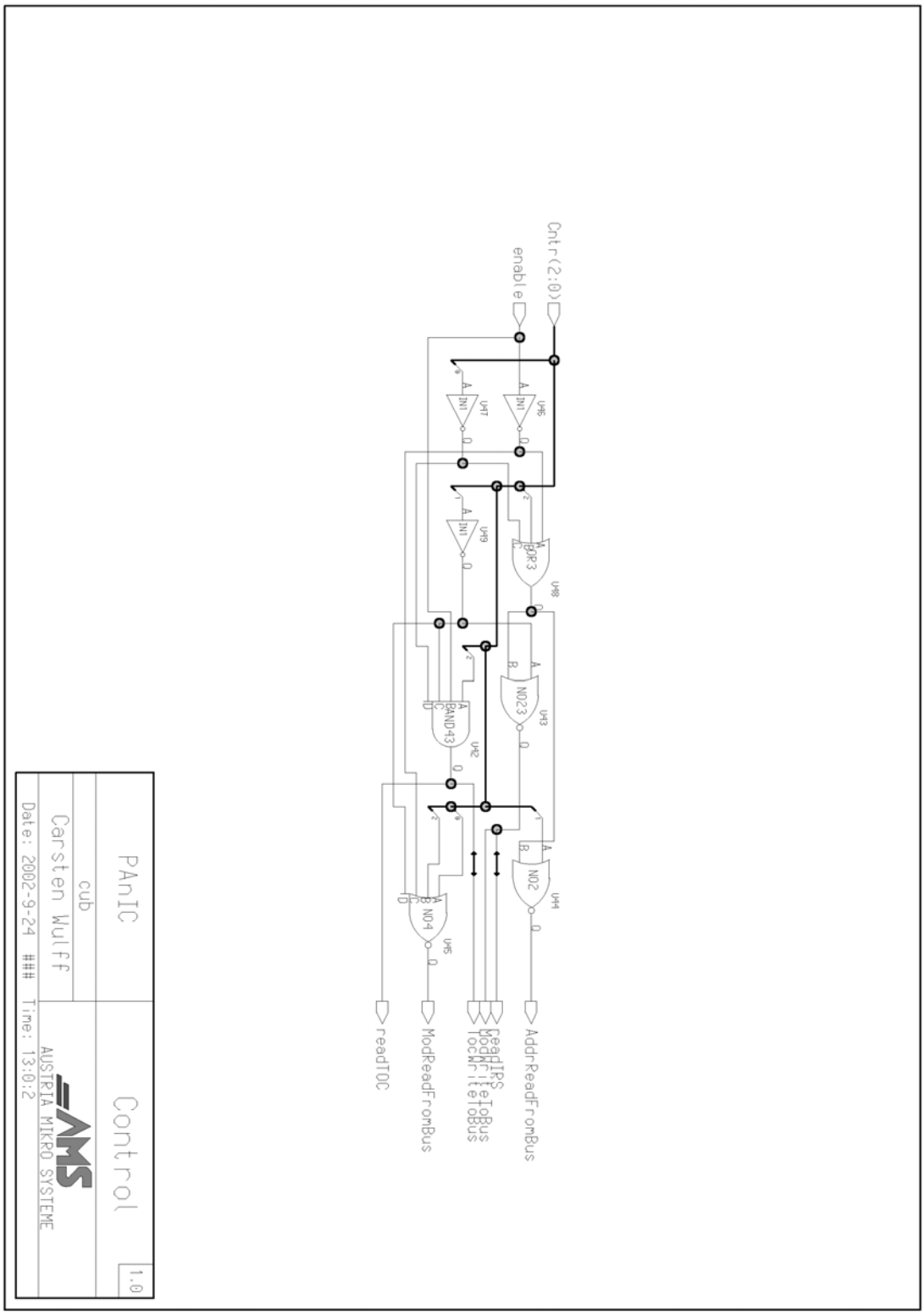

*Schematic 20 IoFallBack*



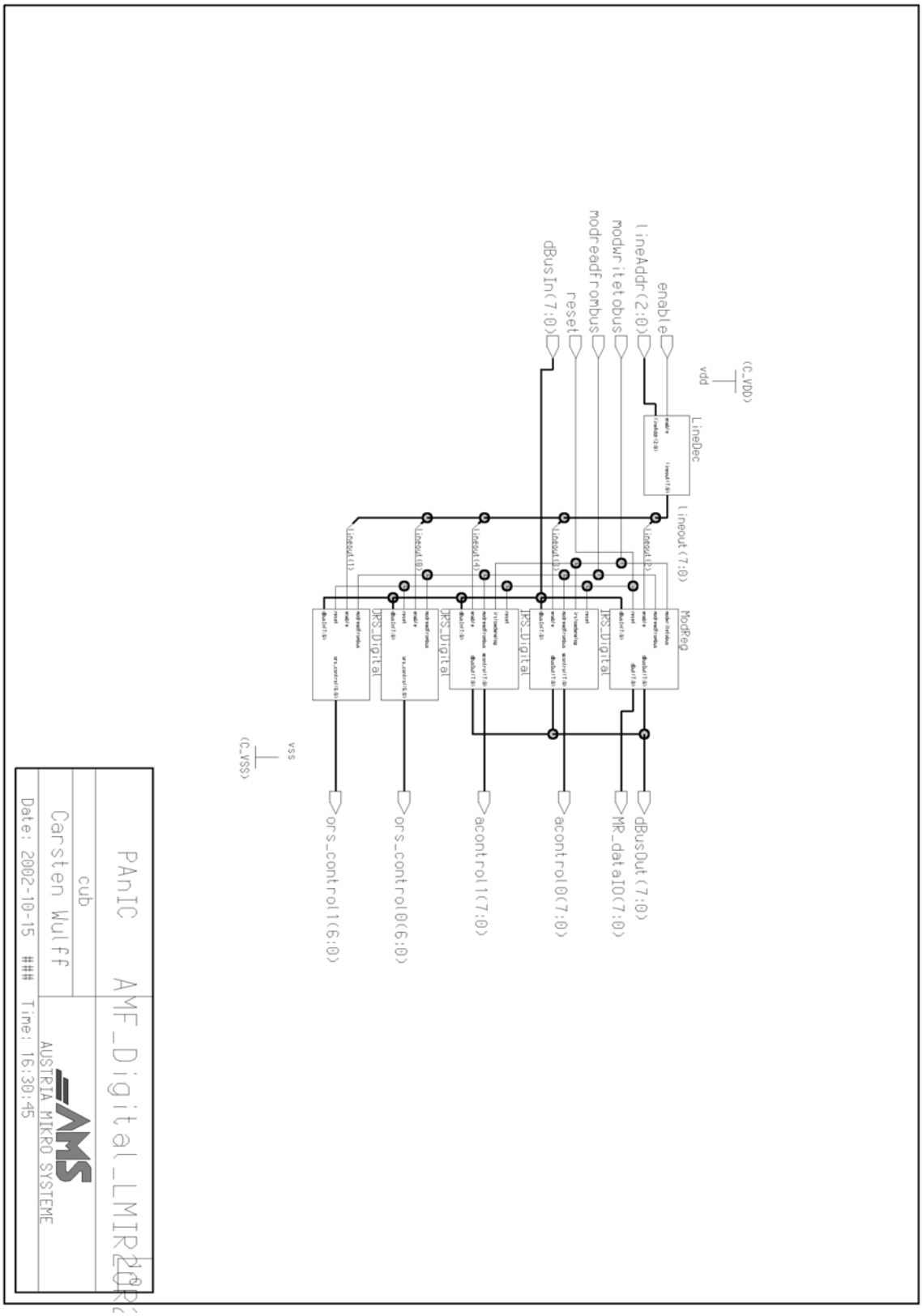

*Schematic 21 Decode4_16*



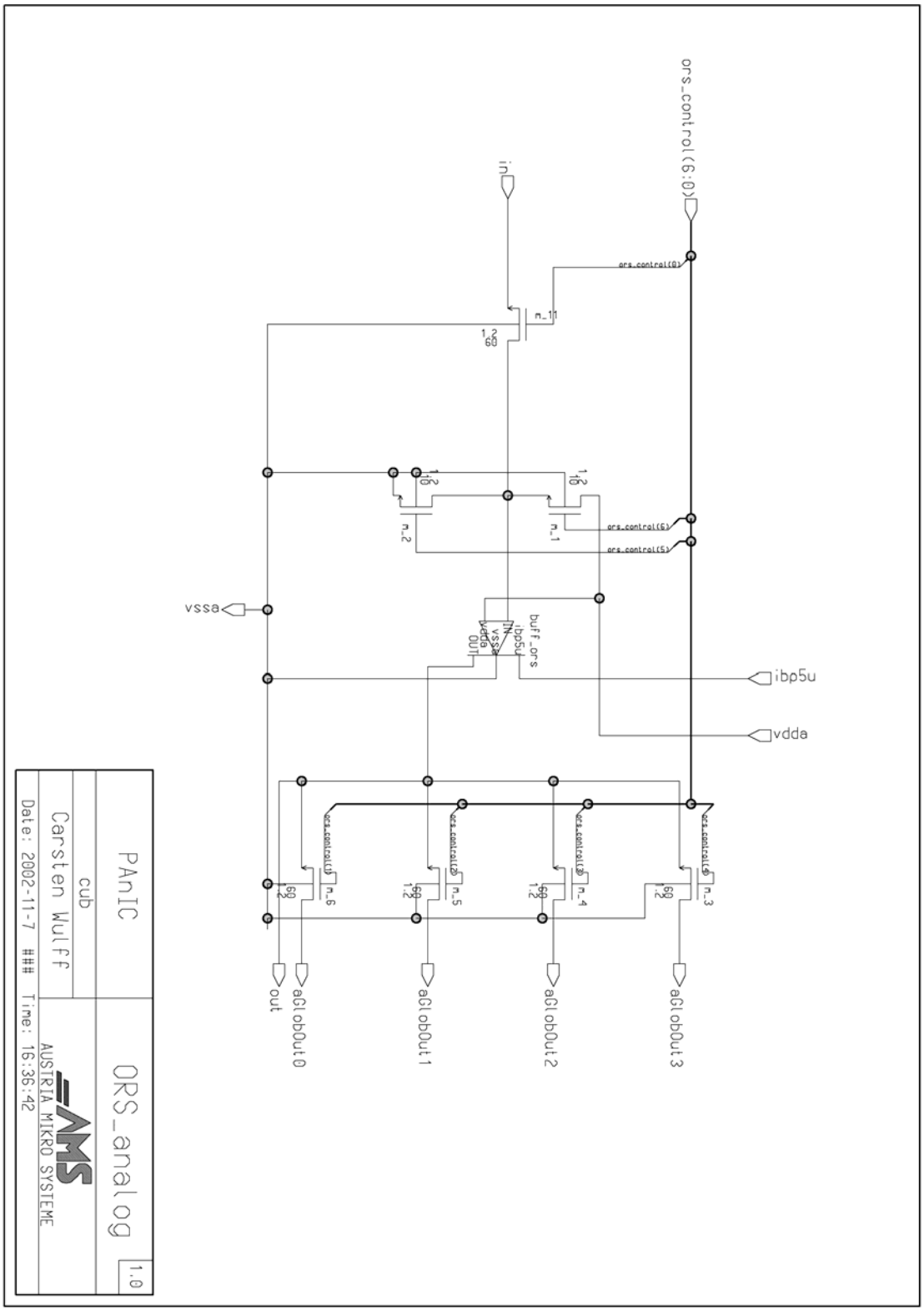

*Schematic 22 SPI*



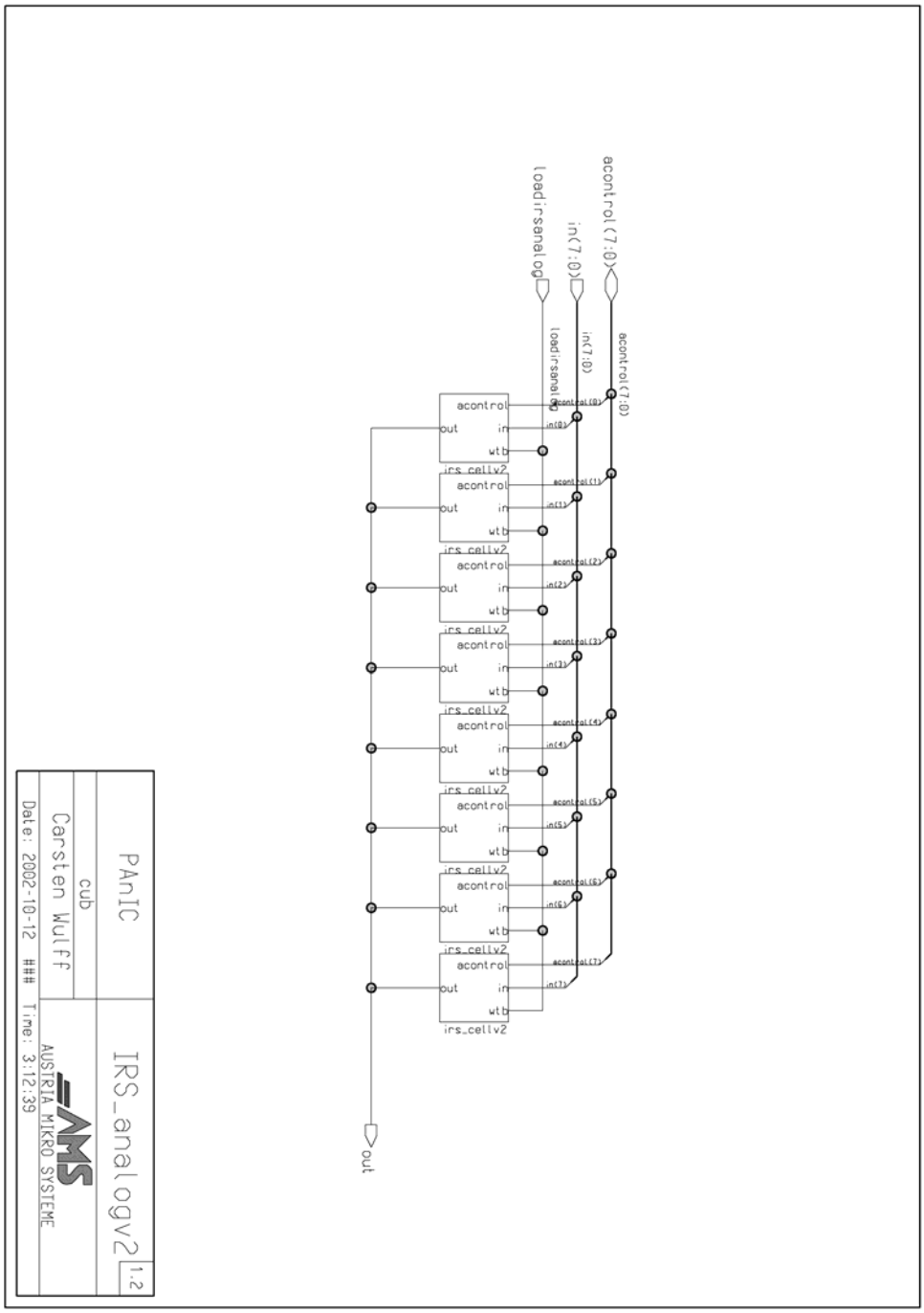

*Schematic 23 Control*



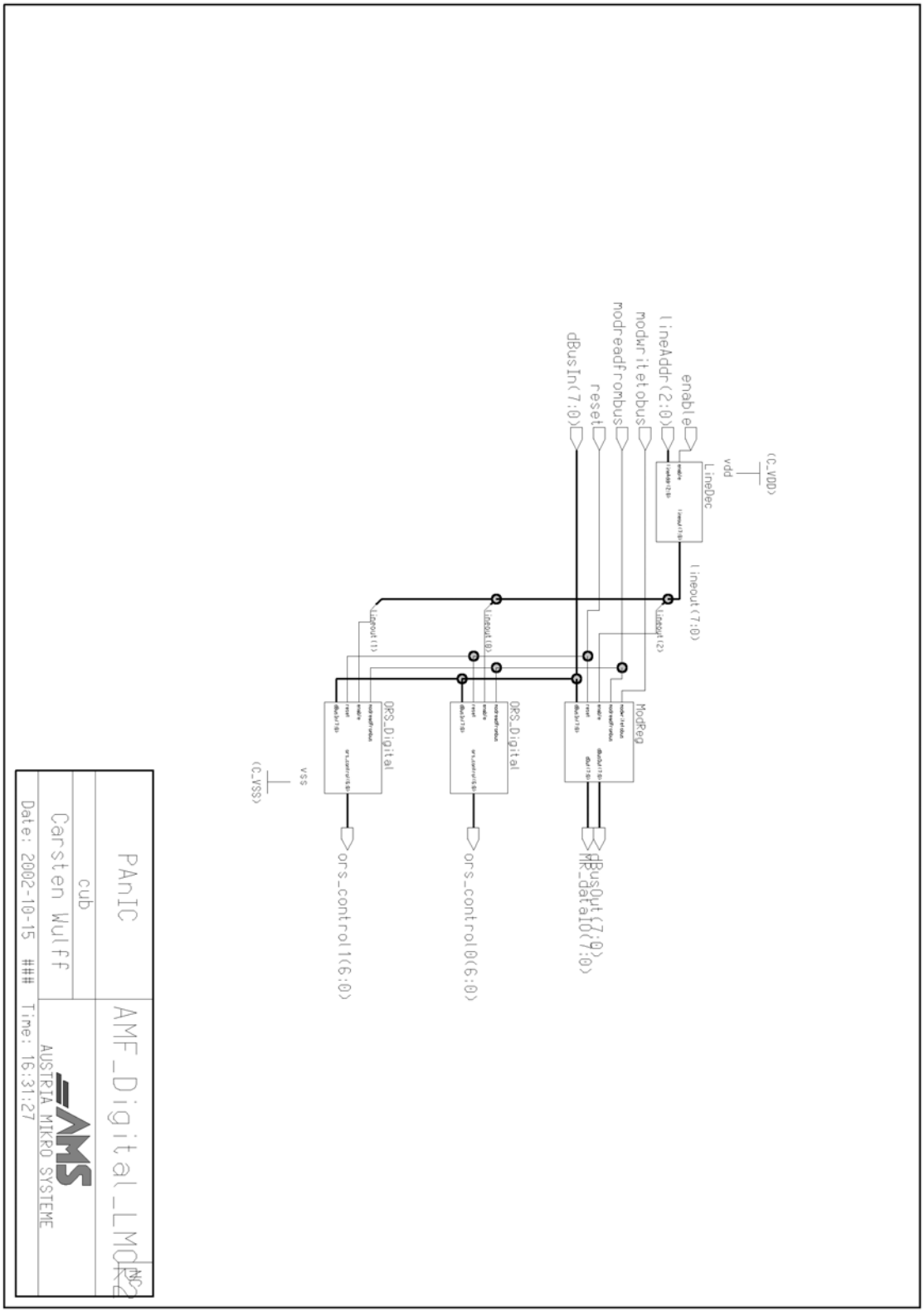

*Schematic 24 AMF_Digital_LMIR2OR2*



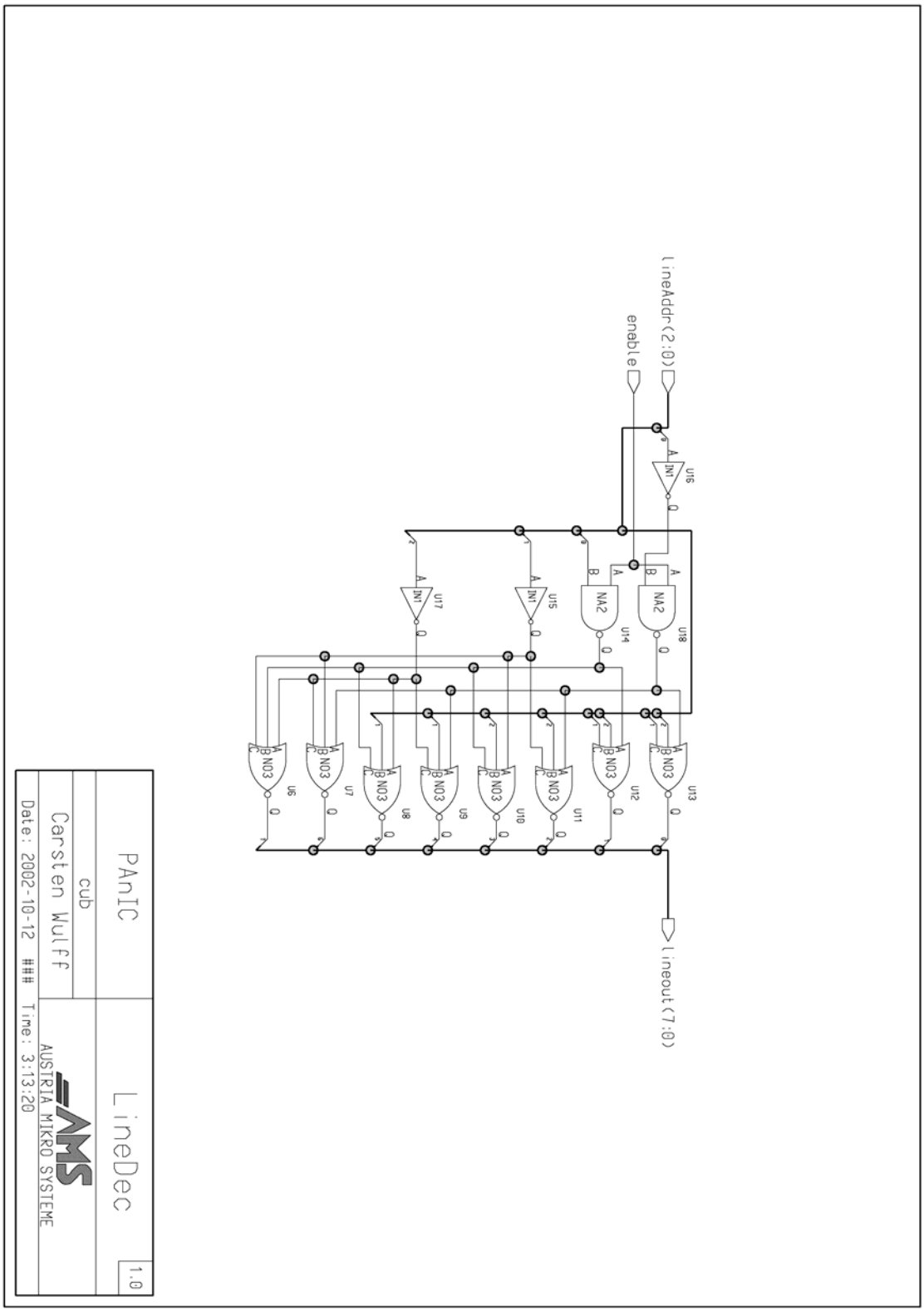

*Schematic 25 ORS_analog*



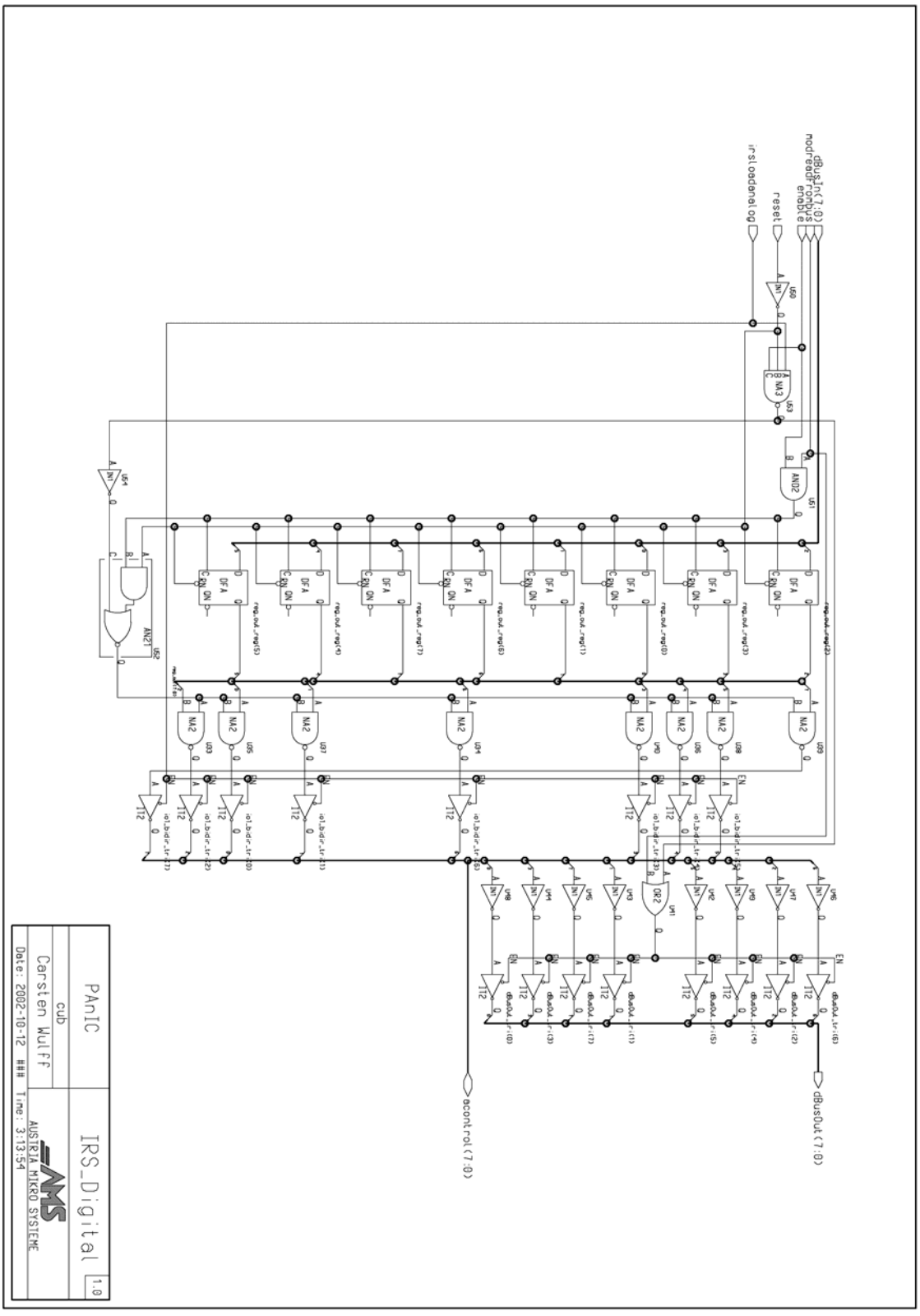

*Schematic 26 IRS_analogv2*



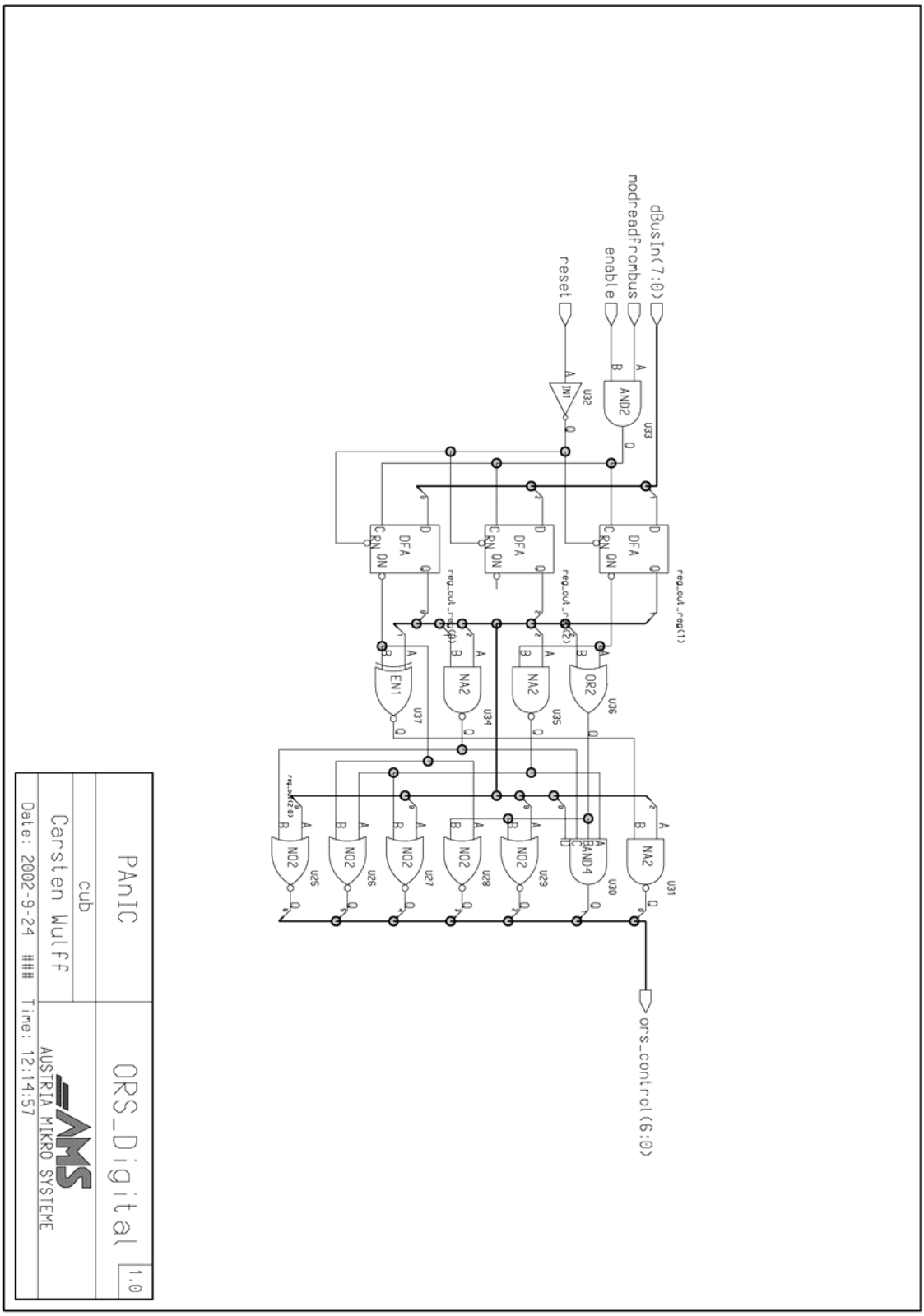

*Schematic 27 AMF_Digital_LMOR2*



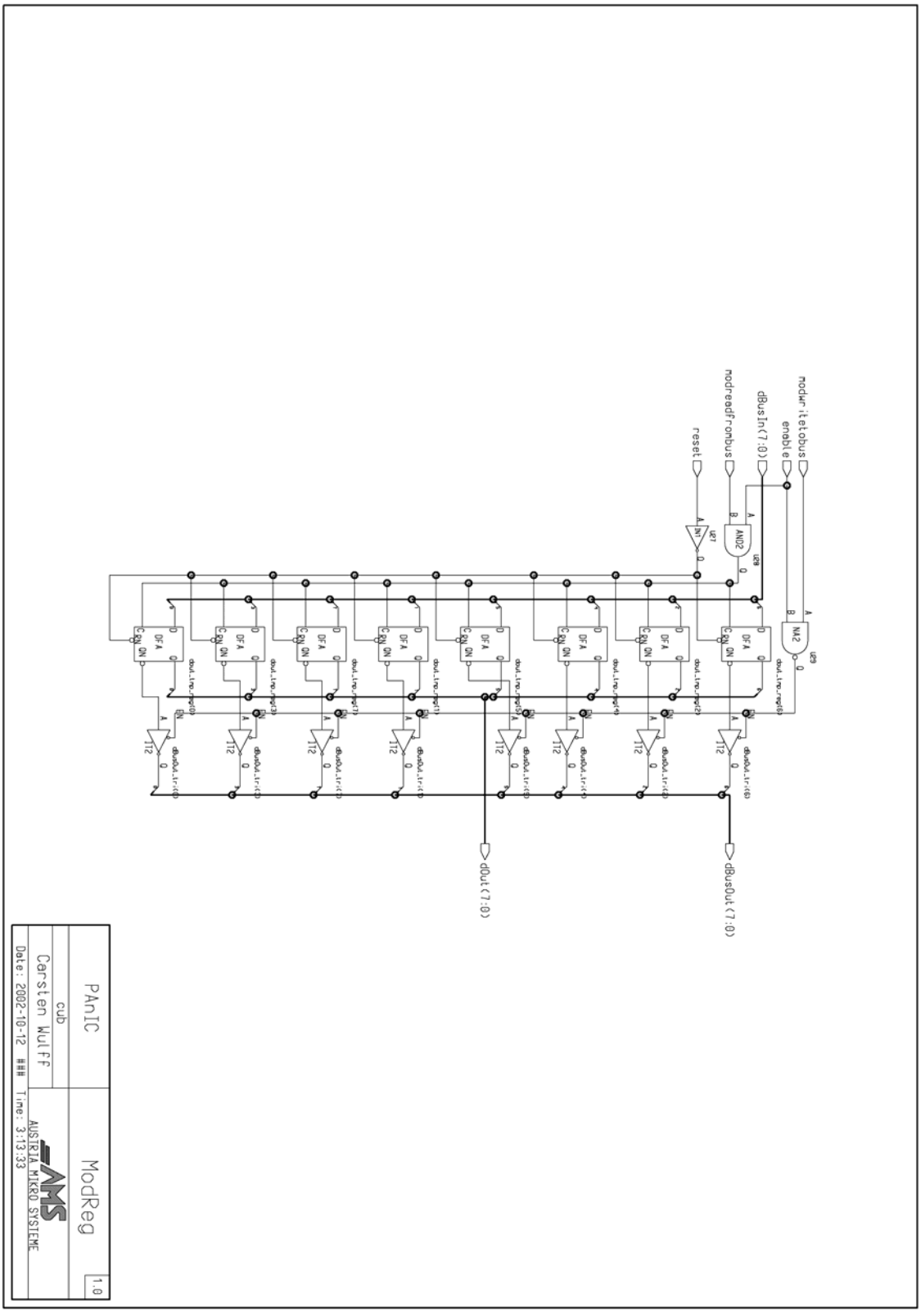

*Schematic 28 LineDec*



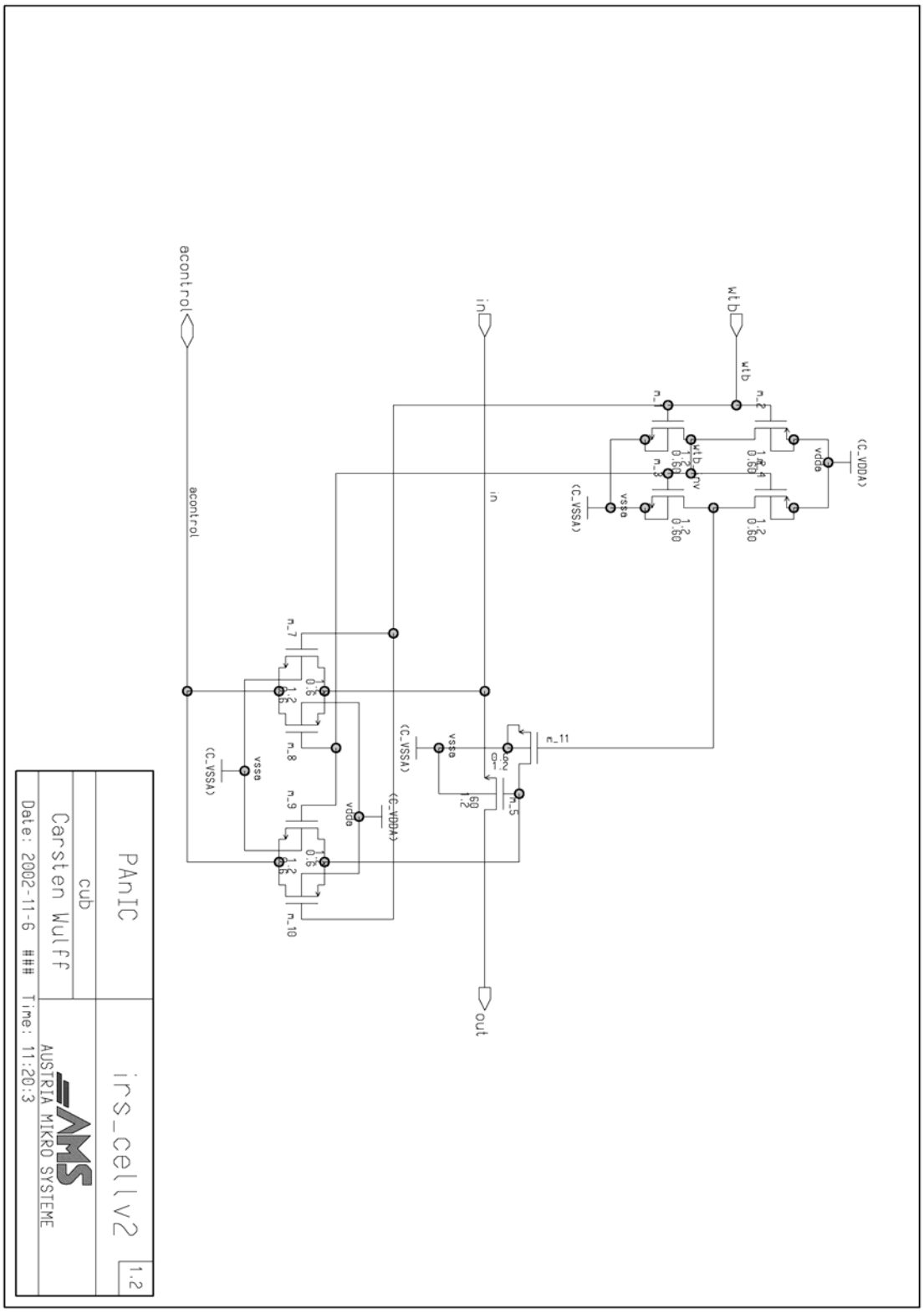

*Schematic 29 IRS_Digital*



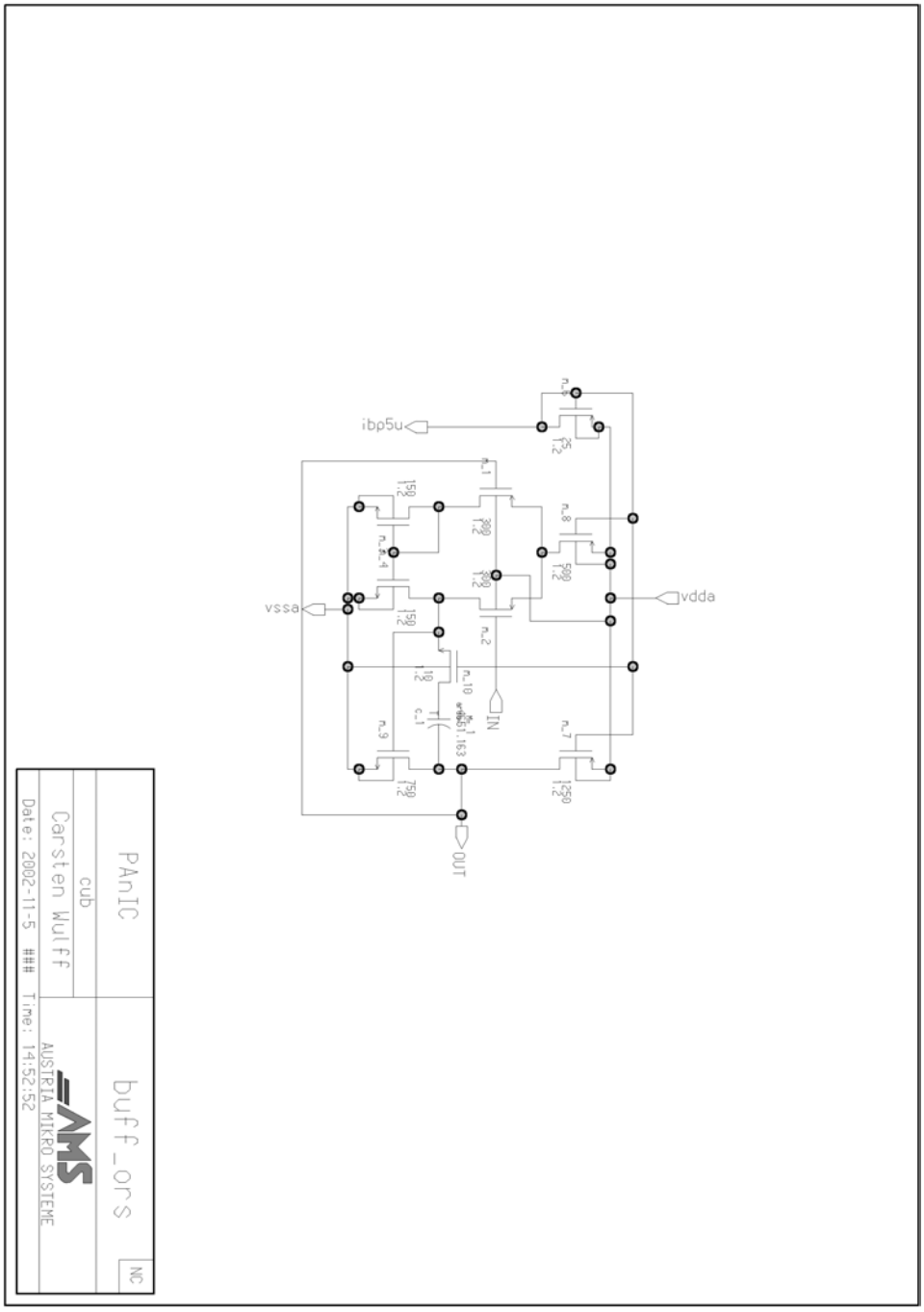

*Schematic 30 ORS_Digital*




*Schematic 31 ModReg*




*Schematic 32 irs_cellv2*




*Schematic 33 buff_ors*

# Appendix IV: LVS

```
 Extraction Errors and Warnings for cell "$USER_WULFF/Diplom/ic/panic/maspan"
------------------------------------------------------------------------------


WARNING:  Short circuit - Different names on one net:
          Net Id:  1
          (1)  name  "PVSSR2"  at location  (3723.65,1756.05)  on layer  35 "MET1"
          (2)  name  "VSS"  at location  (2450.25,3531.65)  on layer  35 "MET1"
          The name "PVSSR2" was assigned to the net.









                ################################################
                ##                                            ##
                ##        C A L I B R E   S Y S T E M         ##
                ##                                            ##
                ##            L V S   R E P O R T             ##
                ##                                            ##
                ################################################



REPORT FILE NAME:         /homes/brutus/wulff/Diplom/ic/lvs.rep
LAYOUT NAME:              $USER_WULFF/Diplom/ic/panic/maspan
SOURCE NAME:              $USER_WULFF/Diplom/da/panic/maspan/vpt_cub_device
LVS MODE:                 Mask
RULE FILE NAME:           /dak2/ams2/mentor/ic_station/cub/rules/cub.rules
CREATION TIME:            Thu Nov 14 19:54:40 2002
CURRENT DIRECTORY:        /homes/brutus/wulff/Diplom/ic
USER NAME:                wulff


*****************************************************************************************************************
                              OVERALL COMPARISON RESULTS
*****************************************************************************************************************


                       #       ###################       _   _
                      #        #                 #       *   *
                   #   #       #     CORRECT     #         |
                    # #        #                 #       \___/
                     #         ###################


  Warning:  Unbalanced smashed mosfets were matched.
  Warning:  Ambiguity points were found and resolved arbitrarily.

-----------------------------------------------------------------------------------------------------------------

INITIAL NUMBERS OF OBJECTS
--------------------------

                Layout    Source         Component Type
                ------    ------         --------------
 Nets:            6191      6190    *

 Instances:       7381      5668    *    mn (4 pins)
                    72         9    *    lddn (4 pins)
                  7405      5650    *    mp (4 pins)
                    23        23         c (2 pins)
                   101       100    *    r (2 pins: p n)
                    30        30         r (3 pins: p n sub)
                  1717       339    *    d (2 pins)
                ------    ------
 Total Inst:     16729     11819


NUMBERS OF OBJECTS AFTER TRANSFORMATION
---------------------------------------

                Layout    Source         Component Type
                ------    ------         --------------
 Nets:            3059      3059

 Instances:        549       549         mn (4 pins)
                     9         9         lddn (4 pins)
                   378       378         mp (4 pins)
                    23        23         c (2 pins)
                    68        68         r (2 pins: p n)
```

```
                    30         30         r (3 pins: p n sub)
                   173        173         d (2 pins)
                  1544       1544         INV (2 pins)
                   348        348         NAND2 (3 pins)
                    10         10         NAND3 (4 pins)
                    75         75         NAND4 (5 pins)
                   109        109         NOR2 (3 pins)
                    51         51         NOR3 (4 pins)
                     2          2         NOR4 (5 pins)
                    14         14         AOI_2_1 (4 pins)
                    13         13         OAI_2_1 (4 pins)
                     8          8         OAI_2_1_1 (5 pins)
                     8          8         OAI_2_2_1 (6 pins)
                   746        746         SDW2 (3 pins)
                   173        173         SDW3 (4 pins)
                  1085       1085         SUP2 (3 pins)
                     2          2         SMN2 (4 pins)
                ------     ------
 Total Inst:      5418       5418


       * = Number of objects in layout different from number in source.


*****************************************************************************************************************
                                        LVS PARAMETERS
*****************************************************************************************************************


o LVS Setup:

   Component Type Properties:             lvs_device spicemodel
   Subtype Property:                      mdl_prim
   Pin Name Properties:                   phy_pin
   Power Net Names:                       VDD VDDA PVDDR1 PVDDR2 vdda3v
   Ground Net Names:                      VSS VSSA PVSSR1 PVSSR2 PVSSR3 PVSSR4
   Ignore Ports:                          YES
   Check Port Names:                      NO
   All Capacitor Pins Swappable:          NO
   Reduce Series Mos Transistors:         NO
   Reduce Parallel Mos Transistors:       YES
   Reduce Semi-Series Mos Transistors:    NO
   Recognize Gates:                       ALL
   Reduce Split Gates:                    YES
   Reduce Parallel Bipolar Transistors:   YES
   Reduce Series Capacitors:              YES
   Reduce Parallel Capacitors:            YES
   Reduce Series Resistors:               YES
   Reduce Parallel Resistors:             YES
   Reduce Parallel Diodes:                YES
   Unused Device Layout Filter Options:
   Unused Device Source Filter Options:
   Soft Substrate Pins:                   NO
   LVS Report Options:
   Expand Unbalanced Cells:               YES
   Globals Are Ports:                     YES
   Reverse WL:                            NO
   Preserve Parametrized Cells:           NO
   Spice Prefer Pins:                     NO
   Spice Slash Is Space:                  YES
   Spice Allow Floating Pins:             YES
   Property Resolution Maximum:           32
   Signature Maximum:                     None
   Layout Case:                           NO
   Source Case:                           NO
   Compare Case:                          NO
   Report List Limit:                     999


o Numeric Trace Properties:

   Component      Component      Source             Direct             Mask               Tole-  Trace
   Type           Subtype        Property Name      Property Name      Property Name      rance

   mn                            instpar(w)         w                  w                    1%    YES
   mn                            instpar(l)         l                  l                    1%    YES
   mp                            instpar(w)         w                  w                    1%    YES
   mp                            instpar(l)         l                  l                    1%    YES
   lddn                          instpar(w)         w                  w                    1%    YES
   lddn                          instpar(l)         l                  l                    1%    YES
   lddp                          instpar(w)         w                  w                    1%    YES
   lddp                          instpar(l)         l                  l                    1%    YES
   r                             instpar(w)         w                  w                    1%    YES
   r                             instpar(l)         l                  l                    1%    YES
   c                             dev_area           a                  a                    1%    YES
   c                             peri               p                  p                    1%     NO
```

```
*****************************************************************************************************
                              INFORMATION AND WARNINGS
*****************************************************************************************************


                 Matched    Matched    Unmatched    Unmatched    Component
                  Layout     Source       Layout       Source    Type
                 -------    -------    ---------    ---------    ---------
   Nets:            3059       3059           0            0

   Instances:        549        549           0            0    mn(nmos4)
                       9          9           0            0    lddn(nmosh)
                     378        378           0            0    mp(pmos4)
                      23         23           0            0    c(cpolya)
                      68         68           0            0    r(rpoly)
                      30         30           0            0    r(rdiffp3)
                      67         67           0            0    d(nd)
                      41         41           0            0    d(nwd)
                      65         65           0            0    d(pd)
                    1544       1544           0            0    INV
                     348        348           0            0    NAND2
                      10         10           0            0    NAND3
                      75         75           0            0    NAND4
                     109        109           0            0    NOR2
                      51         51           0            0    NOR3
                       2          2           0            0    NOR4
                      14         14           0            0    AOI_2_1
                      13         13           0            0    OAI_2_1
                       8          8           0            0    OAI_2_1_1
                       8          8           0            0    OAI_2_2_1
                     746        746           0            0    SDW2
                     173        173           0            0    SDW3
                    1085       1085           0            0    SUP2
                       2          2           0            0    SMN2
                 -------    -------    ---------    ---------
   Total Inst:      5418       5418           0            0


o Statistics:

   4055 layout mos transistors were reduced to 522.
     3533 mos transistors were deleted by parallel reduction.
   4 source mos transistors were reduced to 2.
     2 mos transistors were deleted by parallel reduction.

   66 series layout resistors were reduced to 33.  33 connecting nets were deleted.
   64 series source resistors were reduced to 32.  32 connecting nets were deleted.

   1637 parallel layout diodes were reduced to 93.
   173 parallel source diodes were reduced to 7.

   1 net was matched arbitrarily.


o Layout Names That Are Missing In The Source:

   Nets:         VDDA PVSSR2


o Initial Correspondence Points:

   Nets:         VDD cntr(1) cntr_io(1) cntr(0) cntr(2) cntr_io(0) panic_out(1) panic_in(1)
                 panic_out(5) panic_in(5) panic_out(0) panic_in(3) panic_in(0) panic_in(2)
                 panic_out(3) panic_in(7) panic_in(4) panic_out(7) panic_out(6) panic_out(2)
                 panic_in(6) panic_out(4)


o Ambiguity Resolution Points:

      (Each one of the following objects belongs to a group of indistinguishable objects.
       The listed objects were matched arbitrarily by the Ambiguity Resolution feature of LVS.
       Arbitrary matching may be prevented by assigning names to these objects or to adjacent nets).

       Layout                                                Source
       ------                                                ------

                                    Nets
                                    ----

       1345(3078.500,1559.450)                               /I$31/N$1266



o Matched Mosfets Which Have Been Unequally Reduced:
```

```
       3259(1666.150,2325.150)                                  /I$31/I$1/I$821/I$7464/m_5
       3260(1666.150,2332.350)                                  /I$31/I$1/I$821/I$7464/m_2
       3261(1691.150,2340.750)                                  ** missing smashed mosfet **

       3158(1509.350,2342.550)                                  /I$31/I$1/I$821/I$7465/m_5
       3175(1509.350,2326.950)                                  /I$31/I$1/I$821/I$7465/m_2
       3176(1509.350,2334.150)                                  ** missing smashed mosfet **



*************************************************************************************************************
                                        SUMMARY
*************************************************************************************************************

Total CPU Time:      246 sec
Total Elapsed Time:  249 sec
```

# Appendix V: Excerpt from PAIC project report

*(Chapter 3 DESIGN)*

This chapter will explain the system level design of the PAIC and how the final concept came to life.

## Interface

When choosing the data interface for PAIC there were two key considerations: it should be simple to use and have minimal impact on the number of pins. The second consideration resulted in choosing a serial interface. There are a number of serial interfaces available, i.e. Universal Asynchronous Receiver and Transmitter (UART) or Serial Peripheral Interface (SPI). Both UART and SPI are widely supported in micro-controllers on the market, but SPI simpler to implement. Therefore SPI was chosen as the PAIC data interface. An explanation of the SPI interface is given in Appendix I.

## Version 1

The first version of the PAIC routing network is based on the assumption that there will be a limited number of nodes in an analog circuit. Therefore, there is no need to provide all possible connections of analog cells. It is also based on the assumption that the analog cells will have different number of ports, and thus it does not distinguish between inputs and outputs. An overview is provided in figure 1, we can see the control and routing blocks from Figure 1. An example of a signal highway is provided in figure 2.

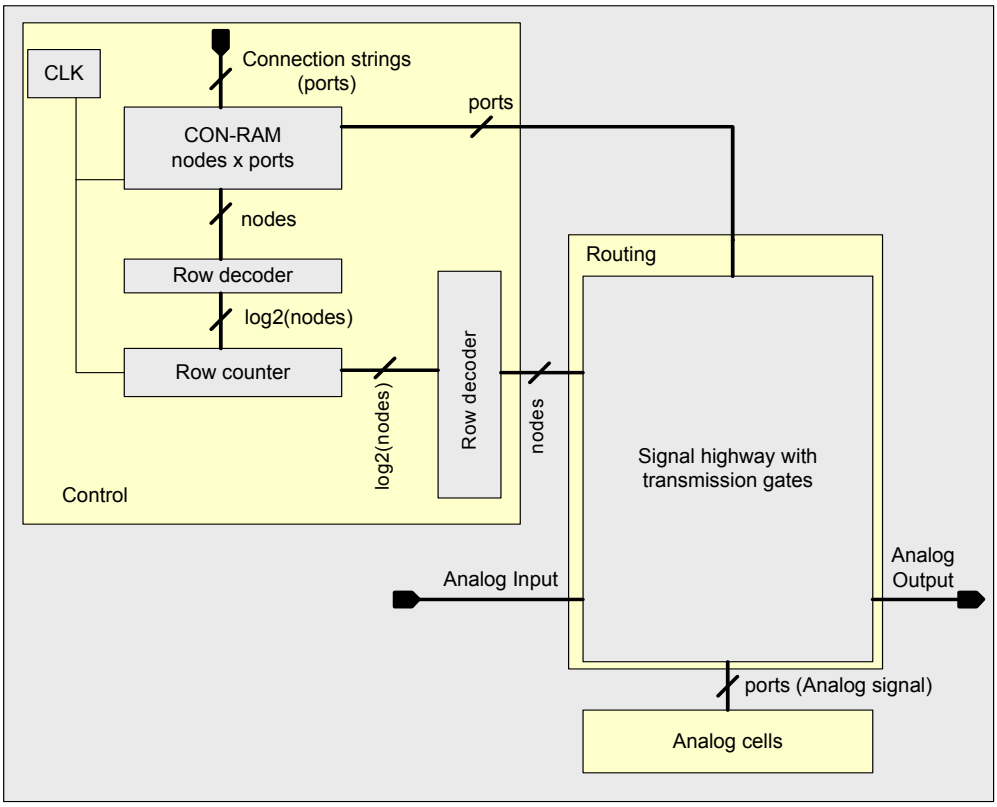



*Figure 1. PAIC version 1*

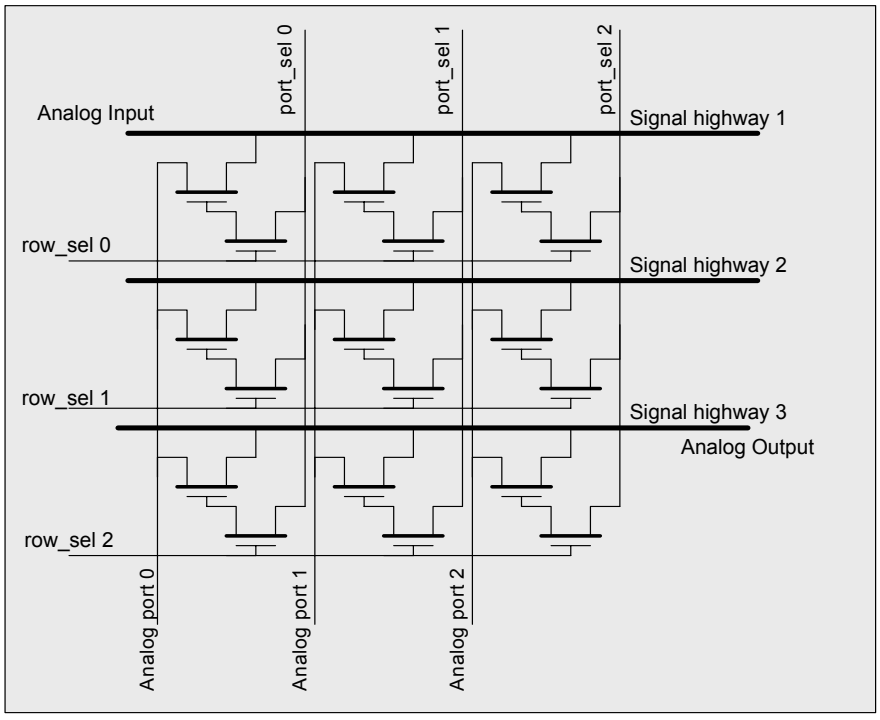



*Figure 2. Signal highway example*

Each of the analog ports (inputs and outputs of the analog cells) are connected to transmission gates which in turn are connected to the signal highway. Controlling the transmission gate is a second transmission gate with connections to the row decoder and the connection RAM. Cycling the RAM and the signal highway simultaneously

updates the signal highway connections. The RAM size is given by the number of ports times the number of nodes (signal highways). In a theoretical circuit this could amount to 16x30 bit RAM, this is based on 10 analog cells with three ports each and a maximum of 16 nodes in a circuit. This results in 480 bits to reprogram the circuit. An example of a circuit is given in figure 3, it uses 4 nodes and 8 ports giving a total 32 bits to reprogram the circuit. Figure 4 shows the equivalent circuit of Figure . The advantage of version 1 is high routing capability, only limited by number of nodes in the analog circuit. The disadvantages are: Complex control of routing network, no readback support, introduces noise because of frequent transitions close to signal highway (row_select transmission gates), analog cell output load dependent on routing network and a need for on-chip RAM

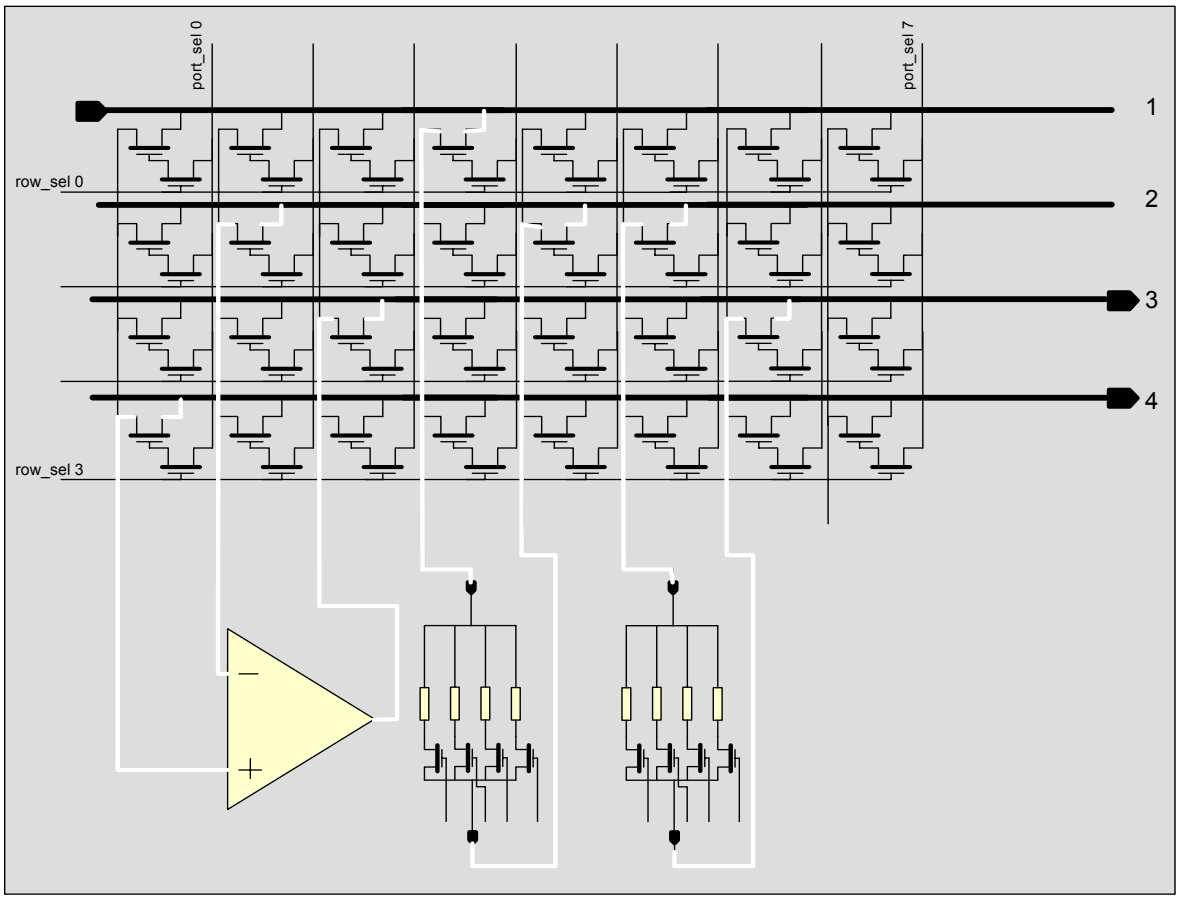



*Figure 3. Example circuit*

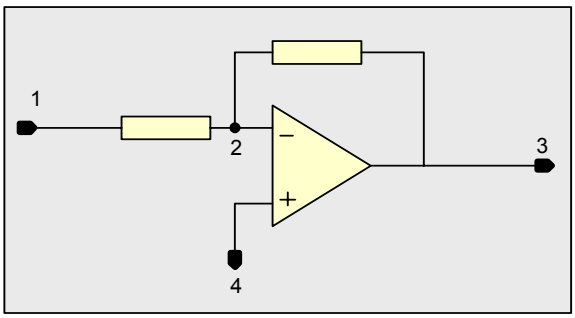



*Figure 4. Equivalent circuit*

For the connection in figure 4 the RAM would contain the following bits:

| | P0 | P1 | P2 | P3 | P4 | P5 | P6 | P7 |
|---|---|---|---|---|---|---|---|---|
| R0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| R1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| R2 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| R3 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

*Table 1 RAM content for figure 4*

## Version 2

Instead of using a single large signal highway it was decided to use one smaller signal highway for each input. This reduces the number of possible connections but it also reduces the complexity of the control circuitry. A block diagram of version 2 is shown in figure 6. Version 2 uses a string (14 bits) for each input, which is loaded into a shift register. This string is from now on called a module packet. The module packet consists of three main fields. The first 4 bits contain the module address, the next 2 bits contain the line address and the last 8 bits contain the highway payload. The highway payload loaded into the input register & switch (IRS) that controls transmission gates, which in turn controls which analog signal should be connected to the respective output of the IRS as pictured in figure 5. Both the module address and the line address increase with log2(n), hence the number of analog cells and number of input/outputs has little effect on the length of the module packet. The highway payload, on the other hand, has significant impact on the length of the module packet.

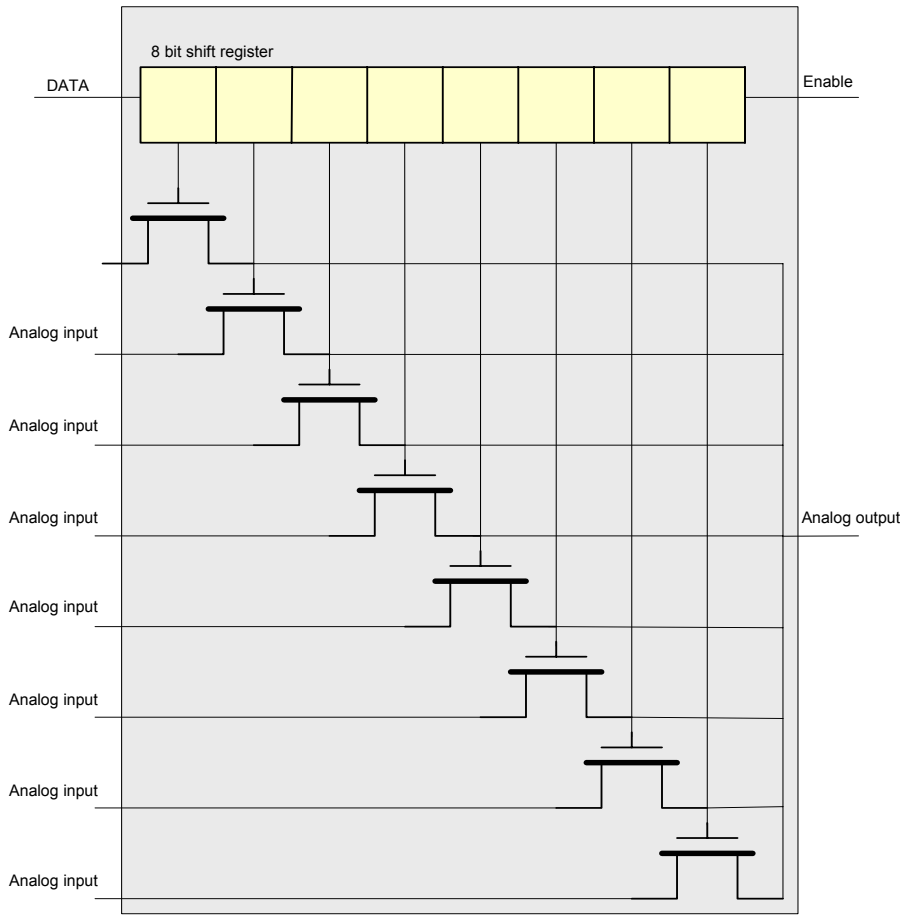



*Figure 5. Input register & switch*

The signal highway in version 2 limits the complexity of the circuits that can be connected. With one IRS per analog cell there is an 8 input limit. One way to circumvent this problem is to use two or more IRS per analog cell. This does not increase the highway payload (HWP) but will increase the line address since the analog framework will contain more than 4 IRS. If 3 bits (8 IRS) where used for the line address the number of possible connections would double (as would the number of clock cycles needed to program the registers). In mathematical terms, n bit increase will give 2^(2+n)*(HWP length) possible connections where as an increase of highway payload will give 2^(2)*(HWP length +HWP length increase). The input to PAIC is a small control circuitry with a SPI interface to the outside world, it controls loading and reading back from the main register and controls a counter which iterates through the highway payload. The highway payload is shifted into the IRS via the counter/mux connection. The module address is connected to a decoder that controls the various framework register enable signals. Version 2 contains an output switch for routing the output signal off chip. The output switch contains a buffer to supply the analog cell with a constant load and to drive the output signals. Version 2 has one global analog input and one global analog output. The advantages of version 2 are: Reduced control complexity from version 1, no need for on-chip RAM and less noise introduced since analog signals can be better separated from digital logic. The

disadvantages are: No readback support, limited input signals to analog cells, few global input/output signals, some noise from the digital portions will still be introduced, longer programming time than version 1.

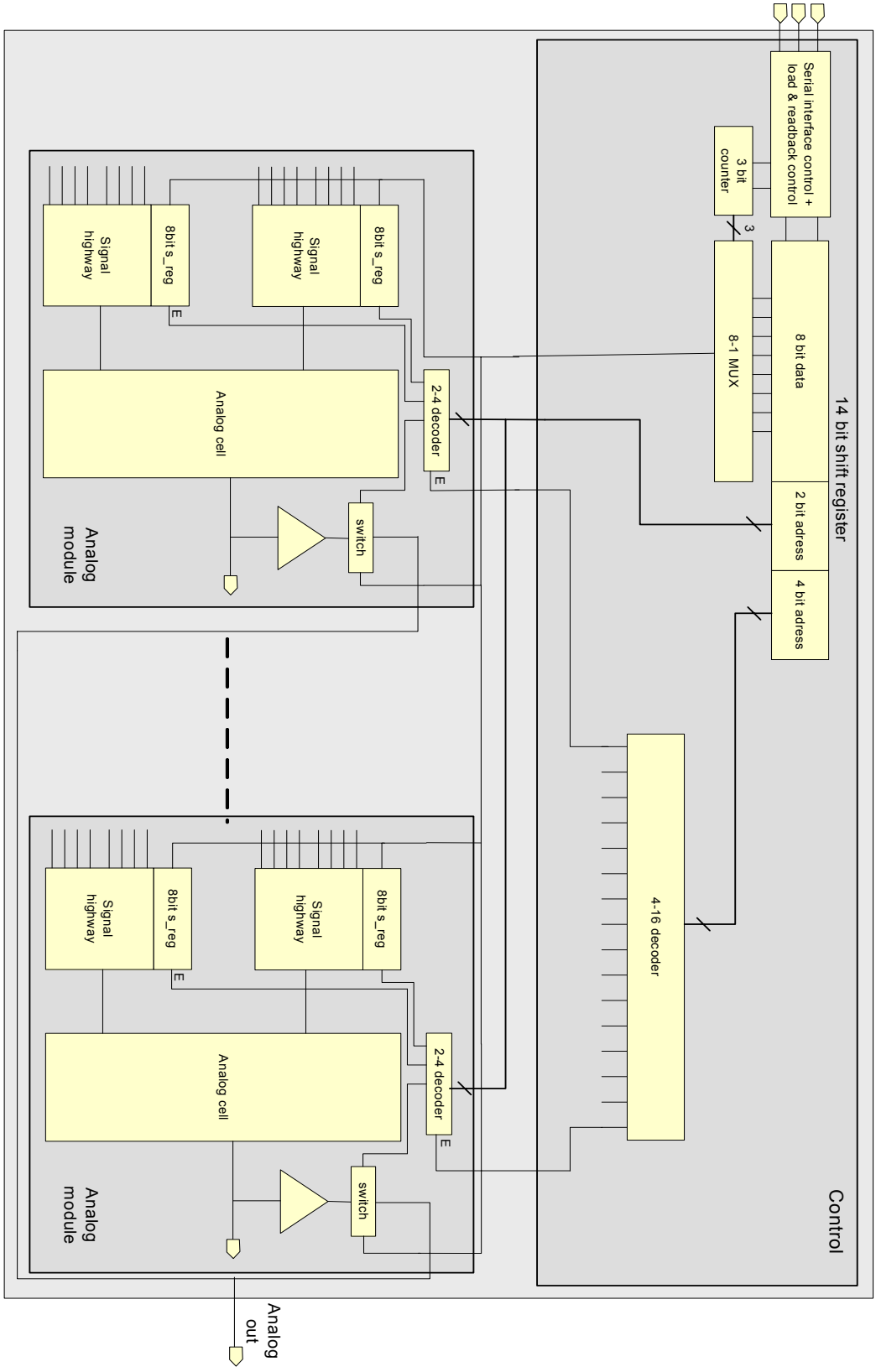



*Figure 6 Version 2 block diagram*

## Version 2 vs version 1

An advantage of version 2 over version 1 is the reduction of noise from the digital modules. In version 2 the analog signal paths can be better separated from noisy digital paths and the input control can open the clock line to the framework registers so the digital circuits close to the analog module are in sleep mode. In version 1 this is not possible since the RAM must be cycled to update the routing network. The greatest advantage of version 2 vs version 1 is the reduction of digital complexity, as always a reduction comes with a cost, but in the version 2 the cost is programming time that can be alleviated by a higher clock rate. Version 2 was chosen as a basis for further research

## Version 3

Version 3 is a modification of version 2 to make it a viable architecture. It has increased the line address to 3 bits and the framework around each analog cell now consists of; a line decoder, five IRS, one internal module register (ModReg) and two output register & switch (ORS). Figure 7 shows an overview of version 3. The IRS is the same as in version 2 with some modification to allow reading the routing network. The module register serves as a control register or a digital input to the analog cell. The ORS is a modification of the output switch in version 2. It can switch the output signal from the analog cell to one of four global outputs and it is modified to allow reading of the routing network. The main register was removed and replaced by an address register (AddrReg). Instead of the shift registers in version 2 all registers in version 3 are parallel load, except a register in the SPI block. Addressing the modules is done through a module address and a line address; both are stored in the 8-bit address register. The module address is connected to a 4-16 decoder (ModDec) that provides an enable signal for the line decoders and the line address is connected to the line decoder (LineDec) that provides an enable signal for IRS, ORS and a module register. An 8-bit bi-directional bus is used for data transfer to and from the registers. Programming and controlling version 3 is done through a SPI for data input/output and a 3-bit bus for control signals. Version 3 also has a table of content, which contains a definition of the analog cells. The master clock was removed from version 3 [12], and the architecture was made semi-asynchronous. Semi-asynchronous involves the use of synchronous registers but not a master clock. The clock signals for the synchronous registers are supplied from the control logic. A thorough explanation of the design will be given later under the implementation section. The advantages of version 3 are: readback support included, more input signals to analog cells than version 2 (line address increase), four global input/output signals, less introduced noise due to no master clock, reduced programming time through the introduction of a global reset signal (no need to program analog modules that will not be used) and an 8-bit digital input to analog cells.

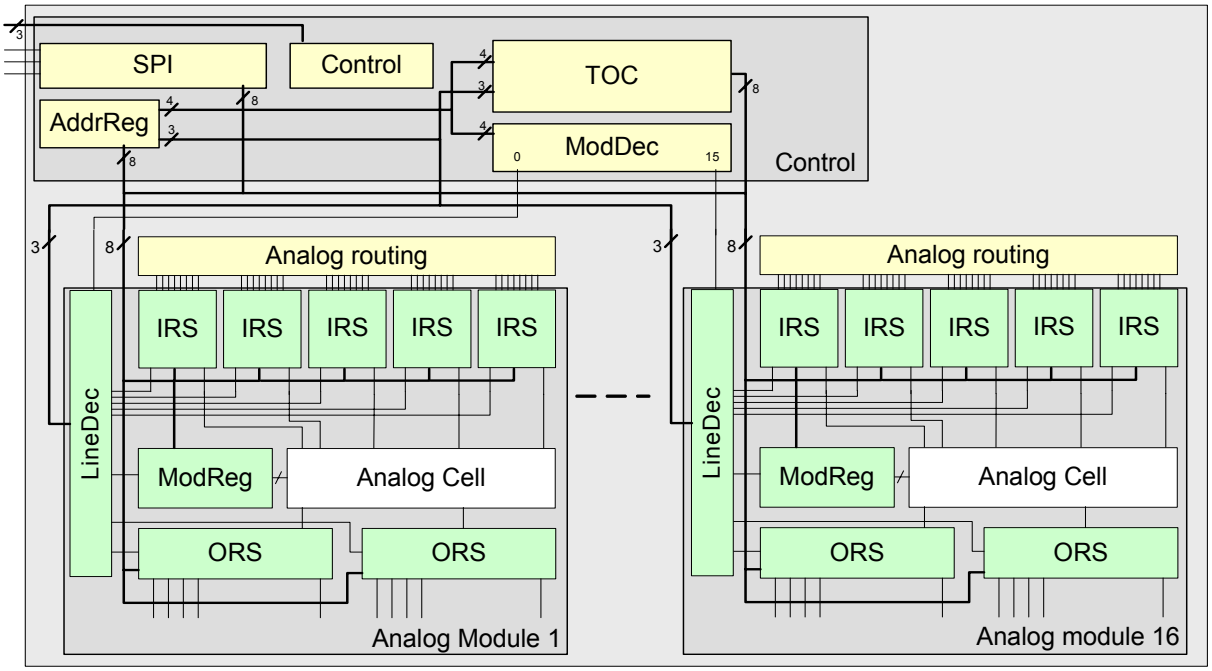



*Figure 7 Overview of Version 3*

## Version 3 vs version 2

Most of the modifications in version 3 were made to make version 2 into a viable architecture i.e. modification of the IRS and ORS to support readback of the routing network. The removal of the master clock signal gives version 3 better noise performance than version 2. Without the master clock there will be no transitions in the digital logic unless data is written to the PAIC, therefore the digital logic will be more quiet than with a master clock.

# APPENDIX VI: NEXT GENERATION LAB – A SOLUTION FOR REMOTE CHARACTERIZATION OF ANALOG INTERGRATED CIRCUITS

Carsten Wulff, Thomas Aas Sæthre, Arne Skjelvan , Trond Ytterdal, Tor A. Fjeldly and Michael Shur
Department of Physical Electronics, Norwegian University of Science and Technology
Email: ytterdal@fysel.ntnu.no

*Abstract*- In this report, we describe the development and use of a remotely operated laboratory based on Microsofts .NET technology. The Next Generation Lab combines the latest in web technology with industrial standard instruments to make a cost effective solution for education in the field of analog CMOS integrated circuits.

## INTRODUCTION

With the mass proliferation of the Internet, interesting possibilities have emerged for extending its use into new areas, including distance-education – a rapidly growing part of the university curricula. By utilizing the WEB, the potential exists for offering courses to remote students, who can participate without other technical requirements than a personal computer and a telephone line.

Laboratory and computer-aided-design modules are vital parts of engineering education, but so far, these elements have been considered impractical for distance-education. On the other hand, user-friendly, computer-controlled instrumentation is revolutionizing the way measurements are being made, and is now permitting net-based techniques to be utilized for setting up remote laboratory access. Such a remote laboratory can, for example, be used in conjunction with courses in electrical engineering, allowing remote students to gain hands-on experience in a wide range of areas. As an added benefit, this technology may offer students the opportunity to work with sophisticated equipment, of the kind they are more likely to find in an industrial setting, and which may be too expensive for most schools to purchase and maintain. Much of the same arguments can be used with regards to software for computer-aided-design.

The basic concept and feasibility of remote system control via the Internet were investigated in two Siv.ing. (M.Sc.) student theses at the Norwegian University of Science and Technology (NTNU) [1], [2]. Further development was pursued in 1998 in collaboration with Professor Shur at Rensselaer Polytechnic Institute (RPI) in Troy, NY. This work has been described in several publications [3]-[11].

So far, the work on the remote laboratories has been dedicated to semiconductor device characterization. It includes several experiments that are performed on a microelectronic test chip, and is used as a lab module in a course on modelling of semiconductor devices at the senior or first year graduate level. In Norway, this is a course that is presently being taught remotely from UniK,located in Oslo, to students at NTNU in Trondheim. The remote lab is planned to become a permanent part of this course.

The main objective of the work presented in this paper is to bring the remote laboratories to the circuit level by developing a Web based lab devoted to characterization of analog integrated circuits. The laboratory is based at the Department of Physical Electronics, NTNU. 4th year students in the courses Analog CMOS 1 and 2 will use this lab.

## PHYSICAL ARCHITECTURE

Next Generation Laboratory (NGL) is built with three main objectives; scalability, easy to add experiments and real time feedback to the user. Through the use of technologies like web services, which are a framework for remote method calls using Hypertext Transfer Protocol (HTTP) and Simple Object Access Protocol (SOAP), the possibility for a distributed architecture emerges. The NGL has a web service that provides a web interface to GPIB (General Purpose Interface Bus) and DAQ (Data Acquisition) boards on the lab workstation, separating the computer connected to the instruments from the application logic. This makes it possible to use dedicated web servers for the web application and lower cost workstation connected to the instruments. Figure 1 shows an example of a possible architecture. The individual workstations can be connected to one or more device under test (DUT).

The physical architecture of the NGL prototype contains the NGL web server and one lab server. Connected to the lab server are a vector network analyser, a power supply, and a Data Acquisition (DAQ) board. These are in turn connected to the DUT, which is a IC containing 9 operational amplifiers (OPAMP) designed as project work in the Analog CMOS 1 course.

We chose to measure the frequency response of the opamps connected in a closed loop as a prototype experiment. The experiment allows the user to specify closed loop gain, bias current and offset from common mode level at positive input of the opamp. The wiring diagram for the AnCMOS chip is shown in Figure 2.

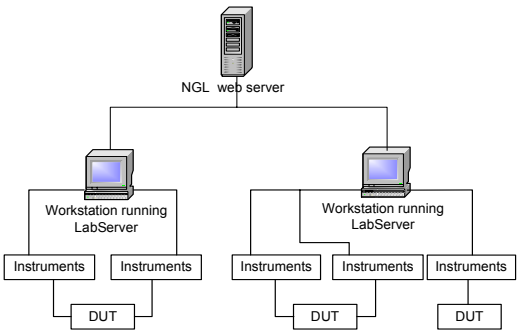



Figure 1 The physical architecture of NGL

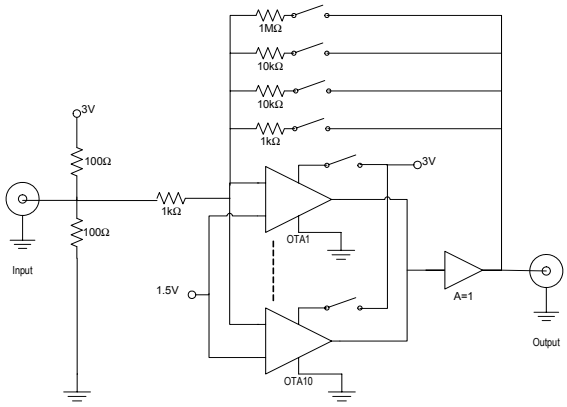



Figure 2 Wiring diagram for the AnCMOS chip

## SOFTWARE ARCHITECTURE

The NGL is based on Microsoft's emerging .NET technology and Scalable Vector Graphics (SVG) from Adobe. The .NET framework has a large class library that was extensively used for the NGL. The NGL application, except for the LabServer and client-side, was written in C#, a modern object oriented programming language. The LabServer was written with a combination of Managed and Unmanaged C++. For the client-side we chose JavaScript. Figure 3 gives an overview over the NGL prototype.

Each of the instruments connected to the DUT is represented in software by a corresponding object. This object provides the basic functionalities of the instrument, it makes use of a proxy object on the web server which is a local representation of the LabServer web service. The fact that the methods for writing GPIB strings and using DAQ commands are located on another computer is thus made transparent for the instrument object

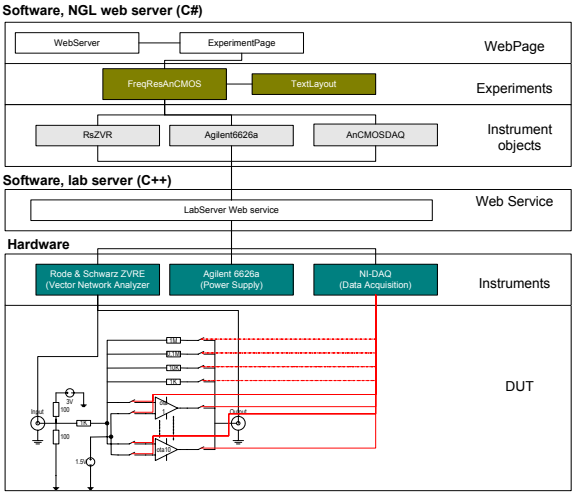



Figure 3 Prototype Overview

The NGL does not only provide scalability in hardware, but also in software. All experiments are implemented as an object in the NGL application. *Experiment* is the base class for all experiment, and all classes that inherit *Experiment* are automatically available through a JavaScript menu on the NGL web application. In C# an object can be casted to the type of it's parent and still retain the specialization of the child. This makes it possible to run-time decide which experiment to run, and in addition provide a common framework for all experiments i.e. the same queue control and XML (Extended Mark-up Language) formatted string to show the result. This of course limits the possibilities for experiment especially in the way results are presented, but the graph engine can handle both linear and logarithmic plots, auto scaling of values from yotta to yocto, sizing of plots and "unlimited" number of plot. The actual number of plots is of course restricted by the space on the web page.

The prototype experiment is of a batch type set-up, the user specifies the parameters and runs the experiment. Running the experiment takes around 300ms and to ensure thread safety the NGL application implements a simple form of queue control. When a user submits the parameters, the web application checks to see if there are other experiments running, if there is no experiments running it takes control and runs the experiment.

When it is finished it frees it's control and allows other experiments to run.

SVG is used to plot the results of the experiment, as mentioned the experiment provides an XML formatted string, this string is parsed by a SVGcontrol object which generates JavaScript that draws the graph.

The web application continuously provides the user with status update by means of writing and flushing the output stream without breaking the connection. This is especially important if the experiment has to wait for access to run.

Adding experiments to NGL architecture follows three simple steps:

1. Write one class that inherits *Experiments*
2.Compile and build into the NGL application
3. Test your experiment

The experiments can in theory be written in VB.NET, C#, PerlNET, C++, J++ or any other language that supports the .NET framework

The NGL is show in Figure 4

## CONCLUSION

The NGL web application provides users with a reliable and efficient tool for analog CMOS integrated circuit experiments. It gives lecturers and students the opportunity to perform real experiments on actual circuits, using industrial standard measurement equipment.

The NGL web application provides a framework for distributed experiment set-ups spanning wide geographical areas.

Choosing ASP.NET as server-side technology provides distributed architecture with no additional cost.

## ACKNOWLEDGEMENTS

The project is funded by the Nordunet project "Internet Technology in Laboratory Modules for Distance-Learning ", principle investigator Prof Tor A. Fjeldly, and the Department of Physical Electronics, NTNU.

## REFERENCES

[1] V. Kristiansen, Remotely operated experiments on electric circuits over the Internet - An implementation using Java, M. Sc. thesis, Norwegian University of Science and Technology (1997).

[2] B. Dalager, Remotely operated experiments on electric circuits over the Internet - Realizing a client/server solution, M. Sc. thesis, Norwegian University of Science and Technology (1998).

[3] H. Shen, Z. Xu, B. Dalager, V. Kristiansen, Ø. Strøm, M.S. Shur, T.A. Fjeldly, J. Lü, T. Ytterdal , "Conducting Laboratory Experiments over the Internet", IEEE Trans. on Education, 42, No. 3, pp. 180-185 (1999).

[4] T.A. Fjeldly, M. S. Shur, H. Shen and T. Ytterdal, "Automated Internet Measurement Laboratory (AIM-Lab) for Engineering Education", Proceedings of 1999 Frontiers in Education Conference (FIE'99), San Juan, Puerto Rico, IEEE Catalog No. 99CH37011(C), 12a2 (1999).

[5] M. Shur, T. A. Fjeldly and H. Shen, "AIM-Lab - A System for Conducting Semiconductor Device Characterization via the Internet", late news paper at 1999 International Conference on Microelectronic Test Structures (ICMTS 1999), Gothenburg, Sweden (1999).

[6] T.A. Fjeldly, M.S. Shur, H. Shen, and T. Ytterdal, "AIM-Lab: A System for -Remote Characterization of Electronic Devices and Circuits over the Internet", Proc. 3rd IEEE Int. Caracas Conf. on Devices, Circuits and Systems (ICCDCS-2000), Cancun, Mexico, IEEE Catalog No. 00TH8474C, pp. I43.1 -I43.6 (2000)

[7] K. Smith, M. Sc. thesis, UniK/University of Oslo (2000).

[8] J. O. Strandman, M.Sc. Thesis, Norwegian University of Science and Technology (2000)

[9] R. Berntzen, M.Sc. Thesis, Norwegian University of Science and Technology (2001).

[10] K. Smith, J.O. Strandman, R. Berntzen, T.A. Fjeldly, M.S. Shur, "Advanced Internet Technology in Laboratory Modules for Distance-Learning," accepted for presentation at the American Society for Engineering Education Annual Conference & Exposition 2001, ASEE'01".

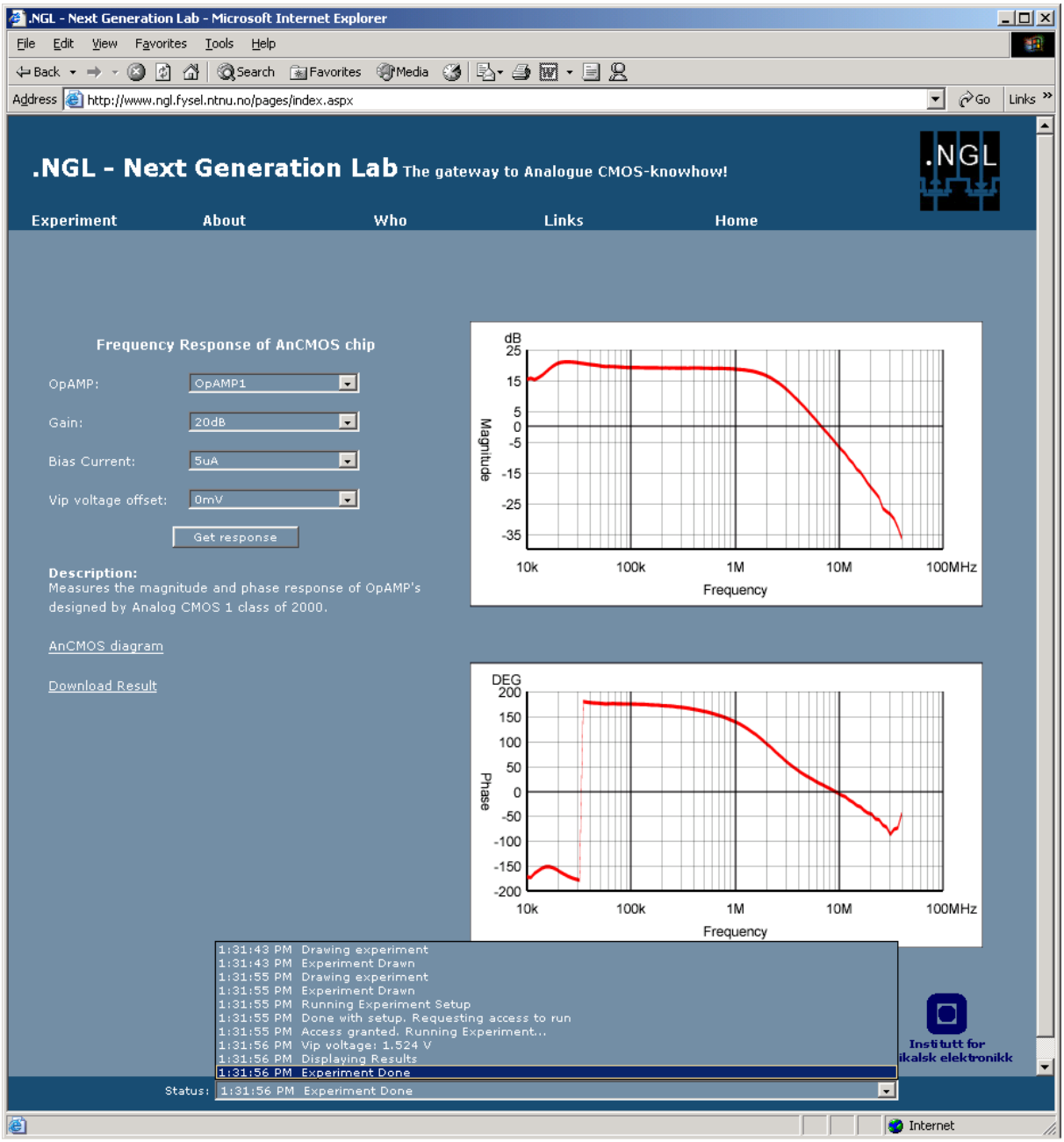



Figure 4 The NGL

# APPENDIX VII: PROGRAMMABLE ANALOG INTEGRATED CIRCUIT FOR USE IN REMOTELY OPERATED LABORATORIES

*Carsten Wulff*[1], *Trond Ytterdal*[2]

**Abstract** — The work presented here aims to outfit remotely operated laboratories with circuit programmability through the use of a programmable analog integrated circuit A concept for remotely operated laboratories using programmable analog integrated circuits is presented. The architecture for a programmable analog integrated circuit and top-level simulations are described.

*Index Terms — Programmable analog integrated circuit, remote laboratory, circuit programmability,*

## INTRODUCTION

Laboratory experience is essential when educating designers of analog (and digital) integrated circuits, providing a base for intuitive understanding of the underlying theory. The test equipment for analog integrated circuits is often expensive, and to equip a lab to serve 30 students is impossible for most universities. Remotely operated laboratories provide a cost advantage in centralizing test equipment while providing students with decentralized concurrent access. Remotely operated laboratories have been explored in [1]-[5], [12]. Already there are remote laboratories that enable a student to make measurements on integrated circuits over the Internet. These laboratories often have a limitation on what types of integrated circuits or devices the student has access to. Some labs [10] have large switching matrixes so the student can select from different circuits to measure, others have a single integrated circuit with some tunable parameters [11], [12]. We propose to take different approach to solve this limitation. Instead of using expensive switching matrixes, we aim to use programmable analog integrated circuits to provide a lab with circuit programmability. We will start by introducing the concept of programmable analog integrated circuits and the system architecture. Then a description of the chip architecture and simulations.

## CONCEPT

We will first explain the concept of programmable analog integrated circuits for those readers unfamiliar with the subject, and then describe the concept for the system architecture.

### Programmable analog integrated circuits

The concept of a programmable analog circuit can simply be described as having an integrated circuit with "standard" cells that can be wired into an analog circuit i.e. a filter or an amplifier. Figure 1 shows an example of a programmable analog circuit. By controlling a routing network, which can connect the analog cells to each other, we can "build" analog circuits. In the figure the two resistors and the operational amplifier are connected together to create an amplifier with gain A=-R2/R1.

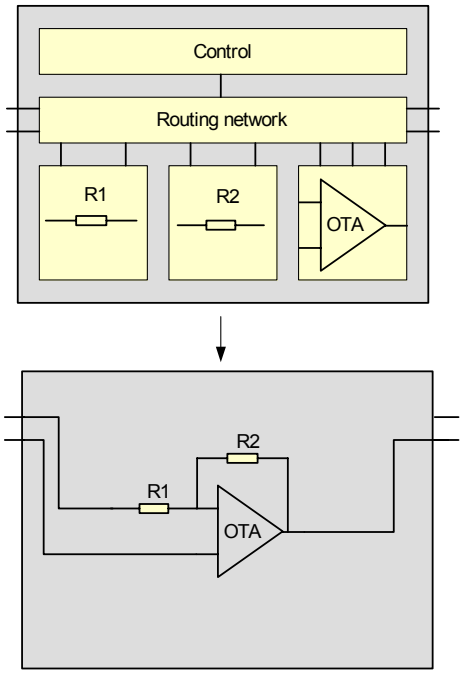



FIGURE 1 PROGRAMMABLE ANALOG INTEGRATED CIRCUIT EXAMPLE

[1] Carsten Wulff, Norwegian University of Science and Technology, Department of Physical Electronics, Trondheim, Norway carsten@wulff.no
[2] Trond Ytterdal, , Norwegian University of Science and Technology, Department of Physical Electronics, Trondheim, Norway ytterdal@fysel.ntnu.no

Programmable analog devices have been reported for more than a decade [6]-[9]. Several manufactures have made programmable analog integrated circuits; among these are Motorola, IPM Inc, Lattice and Anadigm. Several designs of Field Programmable Analog Arrays (FPAA) have been reported [6]-[7], but these are often aimed at a commercial market as an analog counterpart to Field Programmable Gate Arrays (FPGA) for rapid prototyping of analog circuits. The marked for these FPAA circuits has not gained the same momentum as FPGA. This is probably because of the much greater challenges involved in creating a programmable analog integrated circuit. One of the main challenges in creating FPAAs is the fact that analog circuits do not have a smallest common denominator. Digital circuits can (in theory) be created from NAND gates no matter the complexity of the circuit. One approach to circumvent this obstacle is to create expert cells [8]-[9], where each analog cell has a set of tunable parameters i.e. a filter with tunable cut-off frequency. These expert cells are designed by analog designers and are guaranteed to operate within specification. It is a modification of the expert cell approach that has been taken with our Programmable Analog Integrated Circuit (PAnIC).

### System Architecture

An overview of the system architecture is presented in Figure 2 and Figure 3. A web-server is connected to a microcontroller and instruments. The instruments can range from simple multi-meters to expensive network analyzers. The instruments are connected to a circuit board that holds several PAnIC chips. Each PAnIC has a set of analog cells that can be selected alone, or wired together to create a more complex circuit. A student connects to the web-server and gets a graphical user interface that contains a toolbox with the available analog cells. The student draws a circuit from the analog cells in the toolbox and submits the circuit to the web-server. The web-server configures the PAnIC chips, through the microcontroller, to create the circuit the student requested. It then performs measurements on the circuit the student has drawn and returns the result to the student.

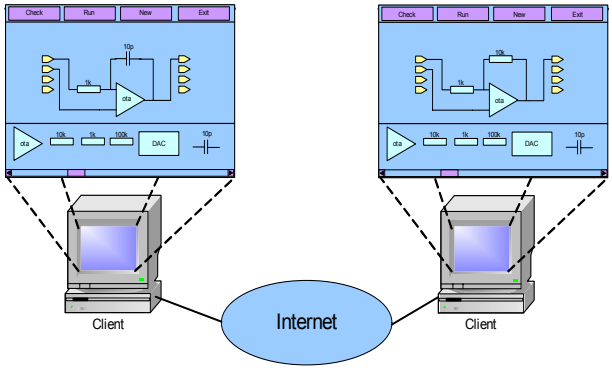



FIGURE 2. SYSTEM ARCHITECTURE: CLIENT

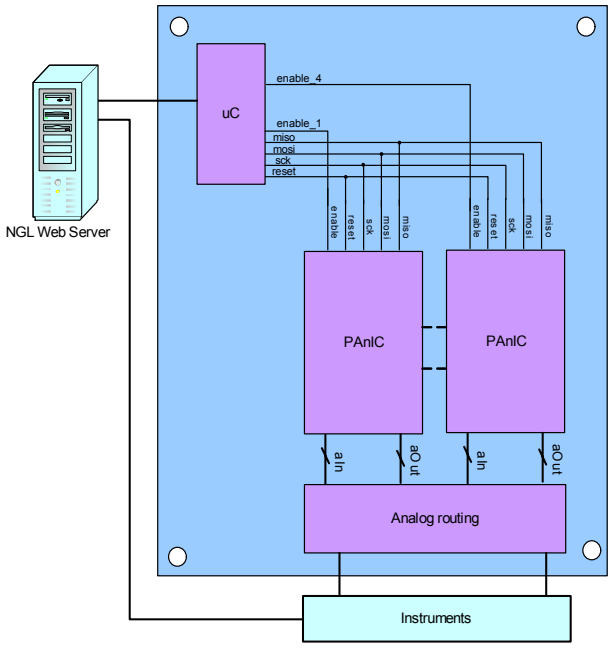



FIGURE 3. SYSTEM ARCHITECTURE: SERVER-SIDE

## PANIC ARCHITECTURE

The PAnIC is based on the ideas published in [8], [9]. The PAnIC consists of control logic, an analog module framework (AMF) and several analog cells. A block diagram is pictured in Figure 4. The PAnIC can be interfaced with most microcontrollers and provides 4 analog input signals and 4 analog output signals.

### Control

Control consists of an serial peripheral interface (SPI), address register, control signal decoder, module address decoder and a table of content. The SPI is used for data and address communication with the microcontroller. The module address register and decoder are for addressing the analog modules ( analog module is the analog cell plus the analog module framework). The PAnIC

architecture can address up to 16 analog modules, but in the prototype we chose to only implement 6. The table of content stores an identification tag for each analog module such that a user does not need to know the address of i.e the differential comparator to use it. When the system starts it can read the table of content and find the module address that correspondes to the desired analog cell.

**Analog module framework (AMF)**

The AMF consists of; input register & switch (IRS), output register & switch (ORS), line decoder and module register. The IRS serves as input for the analog module, each IRS can switch up to 8 signals in any combination and each AMF can have up to 5 IRS cells. In addition the IRS provides a function denoted ReadBack that is essential in the design and will be explained later. The ORS cells provide buffering of the output and can switch the output signal to one (or none) of 4 off-chip output signals. Each AMF has two ORS cells. The module registers servers as an 8-bit digital input and output. It can also be used for control signals for the analog cell. The line decoder provides an enable signal for each of the IRS, ORS and module register.

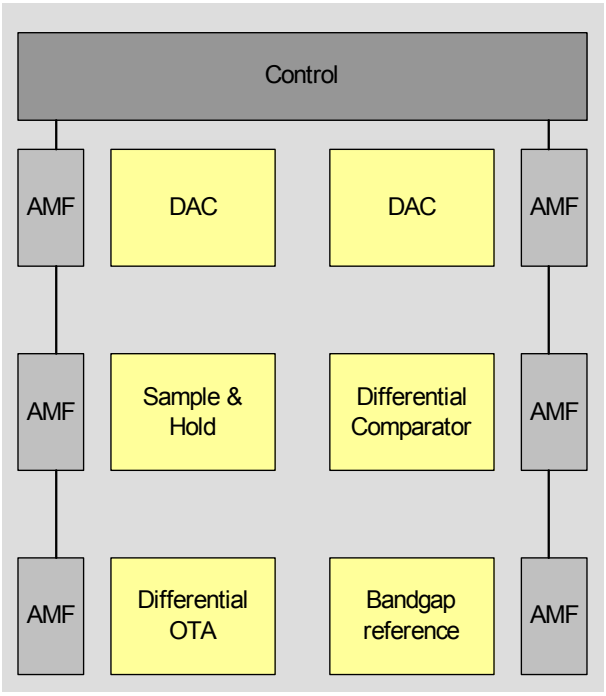



FIGURE 4 PANIC ARCHITECTURE

**Analog Cells**

The analog cells in the prototype consist of a Digital to Analog Converter (DAC), sample & hold, differential comparator, differential operational transconductance amplifier and a bandgap voltage reference. Students in the course Analog CMOS 2, at our university, have designed all analog cells in the prototype. The types of cells are all frequently used in application specific integrated circuits (ASIC). The different ways the cells can be connected is all predetermined during layout of the chip. A possibility for a connection is made by connecting and output signal from an ORS to an IRS of another cell. One of the circuits that can be created with the prototype is an Analog to Digital Converter (ADC). By connecting the DAC, sample & hold and the differential comparator as pictured in Figure 5 we can construct the analog portion of a successive approximation analog to digital converter. An ADC of this type performs a binary search to find the binary output word that most closely resembles the analog input value. It first compares the input value to ½ the signal swing, if the input value is higher the most significant bit (MSB) is 1, if the input is lower MSB is set to zero. It then continues in the half that it knows the input signal lies within, After 8 comparisons the correct digital output word within a resolution of 8 bits is found. The successive approximation register, which controls the binary search, is modeled in the micro-controller.

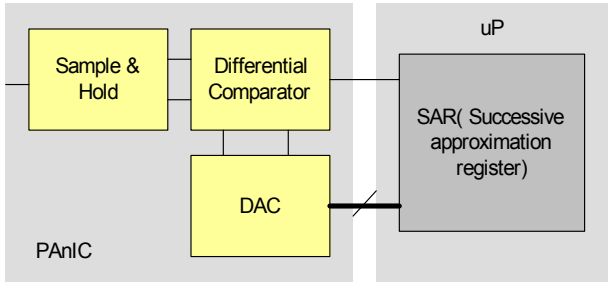



FIGURE 5. ADC BLOCK DIAGRAM

**ReadBack**

ReadBack is, as mentioned, an essential part of the PAnIC design. ReadBack provides a system using the PAnIC chip with a method to discover how to connect the analog cells together and which connections are possible. When the system starts it can request from the PAnIC a "list" of all possible connections between the analog cells. This in combination with the table of content makes the PAnIC a self-consistent programmable analog integrated circuit for use in a flexible environment such as a remote laboratory.

## SIMULATION

A model for the PAnIC was initially constructed in SystemC [13] to test the validity of the concept. Simulation of the PAnIC was done at all levels but especially on circuit programmability, SPI, ReadBack and the Analog to Digital Converter. The simulation results

verified that the concept vas valid. The SystemC model was translated into VHDL for the digital portions, and the analog portions (switches, analog cells and buffers) were modeled in SPICE. The VHDL model was simulated using both a behavioral representation and synthesized netlists using process specific cells. In all simulations, the PAnIC has preformed as expected.

## CONCLUSION

The use of programmable analog integrated circuits in remotely operated laboratories has been introduced. The architecture of PAnIC has been explained and proven through simulations to be a self-consistent programmable analog integrated circuit. The PAnIC provides a remote laboratory with extended flexibility through circuit programmability.

## FUTURE WORK

The PAnIC is in the last stages of the design phase and the prototype will go into production august 2002. A prototype remote laboratory using the PAnIC is scheduled for end of 2002 beginning of 2003.

## REFERENCES


[1] Shen, H. Xu, Z. Dalager, B. Kristiansen, V. Strøm, Ø. et al "Conducting Laboratory Experiments over the Internet", *IEEE Trans. on Education, 42*, No. 3, 1999, pp. 180-18.

[2] Fjeldly, T.A. Shur, M.S. Shen, H. Ytterdal, T., "Automated Internet Measurement Laboratory (AIM-Lab) for Engineering Education", *Proceedings of 1999 Frontiers in Education Conference (FIE'99), San Juan, Puerto Rico*, IEEE Catalog No. 99CH37011(C), 1999, 12a2 .

[3] Shur, M.S. Fjeldly, T. A. Shen, H., "AIM-Lab - A System for Conducting Semiconductor Device Characterization via the Internet", late news paper at 1999 International Conference on Microelectronic Test Structures (ICMTS 1999), Gothenburg, Sweden .

[4] Fjeldly, T.A. Shur, M.S. Shen, H. Ytterdal, T., "AIM-Lab: A System for -Remote Characterization of Electronic Devices and Circuits over the Internet", *Proc. 3rd IEEE Int. Caracas Conf. on Devices, Circuits and Systems (ICCDCS-2000), Cancun, Mexico*, IEEE Catalog No. 00TH8474C, 2000, pp. I43.1 -I43.6

[5] Smith, K. Strandman, J.O. Berntzen, R. Fjeldly, T.A. Shur, M.S., "Advanced Internet Technology in Laboratory Modules for Distance-Learning," *accepted for presentation at the American Society for Engineering Education Annual Conference & Exposition 2001*, ASEE'01.

[6] Lee, E.K.F. Gulak, P.G. "A CMOS field-programmable analog array ". *Solid-State Circuits, IEEE Journal of* , Volume: 26 Issue: 12 , Dec. 1991, pp: 1860 –1867

[7] Gulak, P.G. "Field-Programmable Analog Arrays: Past, present and future perspectives". 1995. IEEE.

[8] Klein, H.W. "Introductory EPAC: an Analog FPGA", *IMP Inc.* ISBN# 0-7803-2636-9

[9] Klein, H.W. "The EPAC Architecture: An Expert Cell Approach to Field Programmable Analog Devices" *Lattice Semiconductor Corp. Analog Integrated Circuits and Signal Processing 17*, 1998, pp 91-103.

[10] Fjeldly, T.A. Berntzen, R. Strandman, J.O. Shur, M.S. Jeppson, K. "LAB-on-WEB" *http://www.lab-on-web.com/*

[11] Ytterdal, T. Wulff, C. Sæthre, T.A. Skjevlan, A., "Next Generation Lab" http://ngl.fysel.ntnu.no

[12] Wulff, C. Ytterdal, T. Sæthre, T.A. Skjevlan, A. Fjeldy, T.A. *et al* "Next Generation Lab – A solution for remote characterization of analog integrated circuits". *International Caracas Conference on Devices, Circuits and Systems (ICCDCS-2002)* .

[13] SystemC, *http://www.systemc.org*