Internship Report

Energy Measurement on the HICANN-X Ultra96 Setup

Heidelberg University, Electronic Vision(s) Group
Supervisor: Yannik Stradmann

Falk Leonard Ebert

January 2021

The HICANN-X Application Specific Integrated Circuit (ASIC) is the most recent implementation of the second generation BrainScaleS accelerated analog neuromorphic computing architecture (BSS-2). A new set of supporting hardware systems, designed for portability and energy efficiency is introduced in this report with a particular focus on the setup's energy measurement capabilities. The hardware and software that enable these measurements is presented along with results of initial energy measurements.

Contents

1 Introduction

The HICANN-X Application Specific Integrated Circuit (ASIC) is the most recent implementation of the second generation BrainScaleS accelerated analog neuromorphic computing architecture (BSS-2) which has been presented in Schemmel et al. (2020). To use this chip in practical neuromorphic computation, supporting hardware and software infrastructure has been developed (Müller et al., 2020) and continues to be improved. In this report, a new set of supporting hardware systems designed to be portable and energy efficient, will be introduced with particular focus on its energy measurement capabilities.

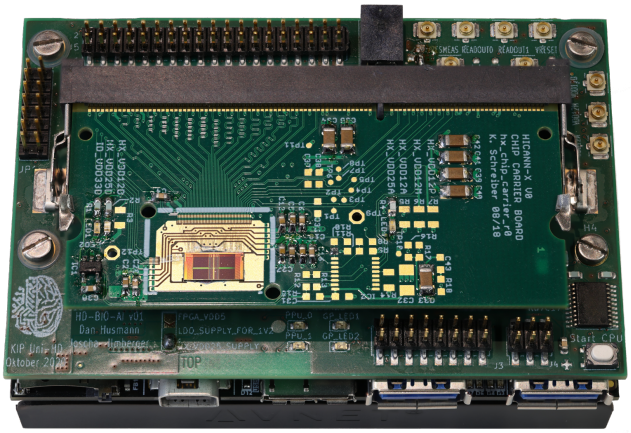

Figure 1: HICANN-X bonded to carrier board inserted into ASICAB on the Ultra96 setup (taken by J. Schemmel)

2 Neuromorphic System Architecture

Neuromorphic accelerators are very complex systems and still the subject of ongoing research, so no plug-and-play solution exists for interfacing with these ASICs. A considerable amount of the work required to build these systems goes into the development of interfacing-hardware and the software framework which are essential in using the capabilities of the neuromorphic hardware.

2.1 Hardware

The HICANN-X neuromorphic accelerator chips, of which one die can bee seen in fig. 1, are highly specialized mixed signal devices with a focus on analog neuromorphic computations. The ASIC offers a generic high-performance interconnect which can be adapted to the workload-specific requirements via an interfacing system with a Field Programmable Gate Array (FPGA) at its heart. This "setup" provides the ASIC with the supply voltages and references it requires, controls the execution of programs and the readout of the results generated by the analog accelerator.

2.1.1 HICANN-X Cube Setup

The established setup for single-chip-usage is the Cube Setup (HX-Cube), which allows experiments to be run on the neuromorphic hardware via host-computers connected over Gigabit Ethernet.
This architecture provides user-friendly remote access to the neuromorphic accelerators with good integration into existing distributed computing infrastructure. The setup's main components will be shortly outlined to allow for better comparison with the new setup later on.

1. **Microcontroller + External I/O Board**
   This board contains a small ARM-Based microcontroller and provides I/O (Ethernet pass-through) for up to 4 FPGA boards. The microcontroller can be accessed via USB to perform maintenance on the FPGA boards like power-cycling or flashing.

2. **K7 Node Board**
   This board contains a *Xilinx Kintex-7* FPGA, which is used to communicate with the neuromorphic chip and auxiliary ICs (Integrated Circuits). The host computer running the experiment control software communicates directly with the FPGA via Ethernet.

3. **xBoard**
   This board is used as an interconnect. It contains voltage regulation circuitry and ICs which allow for analysis of power consumption as well as fine-grained voltage-control for the neuromorphic chip.

4. **HICANN-X Carrier Board**
   This is a small SO-DIMM module which contains the neuromorphic chip itself. It also contains an ID-Chip for electronic identification.

2.2 Software

Since the HICANN-X architecture is capable of accelerating a number of different neuromorphic computing workloads, the software architecture has to provide useful abstraction that provides easy usage for common applications as well as flexibility to accommodate new ways of using the hardware that had not been envisioned previously. This goal is achieved by multiple layers of abstraction, some which are briefly outlined below. This is only a general overview of the architecture to aid in understanding the following sections on software development. A comprehensive explanation of the architecture can be found in Müller et al. (2020).

2.2.1 Hardware Coordinate System

The Hardware Abstraction Layer Coordinate System (*halco*, [3]) is used to address the large number of configuration registers and other components on the system. It provides

a user-friendly interface and reflects the symmetries inherent to the system (Müller et al., 2020).

2.2.2 FPGA Instruction Set

The FPGA Instruction Set Compiler (*fisch*, [4]) abstracts accesses to the systems components (identified by their coordinates) on the basis of *containers*, which can be read from and written to. These container accesses can then be serialized into messages which are understood by the FPGA as described in Müller et al. (2020).

2.2.3 Hardware Abstraction Layer

Building on the aforementioned abstraction layers, the Hardware Abstraction Layer (*haldls*, [5]) provides nested data structures which consolidate the individual hardware configuration registers into logical units of increasing abstraction which are called *containers* (Müller et al., 2020). This allows the user to more easily configure the hardware to suit their needs without expert knowledge of the hardware's inner workings.

3 Ultra96 Setup

The cube setup is not ideal for standalone or embedded use of the chips though, since it is rather large, not very power-efficient and requires a host computer. To improve on these aspects, a new hardware-platform has been developed on the basis of the Avnet Ultra96 development board (Avnet Inc., 2020). It is shown in fig. 1 and its main components are the following:

1. **Ultra96 development board**
   General purpose development board built around a *Xilinx Zynq* MPSoC, which provides a quad core 64bit ARM processor and programmable logic.

2. **ASIC Adapter Board (ASICAB)**
   This board is compatible with the expansion interface of the Ultra96 and the equivalent of the xBoard in the cube setup. It provides power regulation, current reference and general IO circuitry. It also contains the power-measurement ICs, which will be discussed in more detail later.

3. **HICANN-X Carrier Board**
   The same carrier board as in the cube setup is used.

3.1 Design Goals

The Ultra-Setup is primarily designed for energy efficient standalone use. At the same time it is crucial to provide all the same functionality as the cube setups, including compatibility with existing experiments, which require a host-computer with more processing power than the ARM processor can provide. To analyze the power usage and improve

on the system's energy efficiency, the ASICAB contains circuitry to measure the power consumed by the accelerator-chip on its voltage rails, as well as total system power. The chosen development board provides very compact physical dimensions, power usage of under 10 Watts and extensive interfacing options.

3.2 Interfaces

The Ultra96 board provides USB interfaces, which can be used in a number of ways. One important use-case is the connection to a distributed computing cluster via a USB Ethernet adapter to provide compatibility with workloads that were designed for the cube-setup's architecture. Since the ARM processor is running Linux, it is also possible to use the USB ports in other ways, which opens up many possibilities for future applications. Since this setup is also intended as a candidate for embedded usage of the HICANN-X accelerators, it also provides a General Purpose Input Output (GPIO) interface.

3.3 Supply Voltages

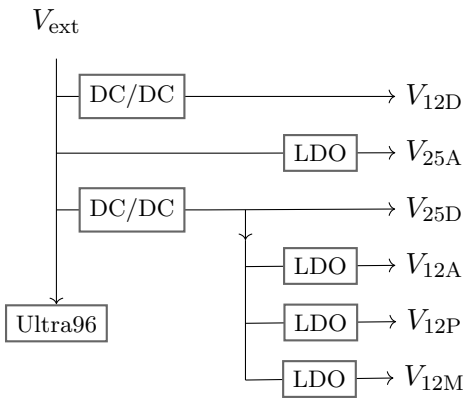

Figure 2: ASICAB power distribution

3.3.1 Voltage Regulation

The ASIC adapter board is supplied with 5 V DC and generates the voltages necessary for the operation of HICANN-X using a combination of DC/DC converters and low-dropout-regulators (LDOs). As can be seen in fig. 2 and fig. 3, power rails with high typical power-draw are regulated using more efficient DC/DC step-down converters. The LDOs for the analog 1.2 V rails are supplied by the 2.5 V rail in order to increase efficiency. Each of these voltage regulators is adjusted by a digital potentiometer to allow for fine-tuning of the voltages and can be turned off via a shutdown signal. The digital potentiometers for voltage adjustment usually don't need to be configured, since

they read their configuration from an on-chip persistent memory automatically. This persistent configuration is written during commissioning of the setup.

Rail	Voltage [V]	Type	typ. Power
V_{ext}	5.0	Input	< 6 W
V_{12D}	1.2	DC/DC	~ 200 mW
V_{25D}	2.5	DC/DC	~ 230 mW
V_{25A}	2.5	LDO	~ 210 mW
V_{12A}	1.2	LDO	< 20 mW
V_{12P}	1.2	LDO	< 20 mW
V_{12M}	1.2	LDO	< 10 mW

Figure 3: Supply voltages generated on the ASICAB. Approximate power-draw during inference in HAGEN mode (see Schemmel et al., 2020, sec. 3.2)

3.3.2 Power Measurement

To analyze the power usage and improve on the system's energy efficiency, the ASICAB contains circuitry to measure total system power and the power consumed by the accelerator-chip on every voltage rail. This is realized using *INA219* power-sensing ICs, that make use of a small shunt resistor which is connected in series with the load. Technical documentation for these ICs can be found in Texas Instruments (2015). By measuring the voltage drop across that resistor it is possible to calculate the current which passes through it.

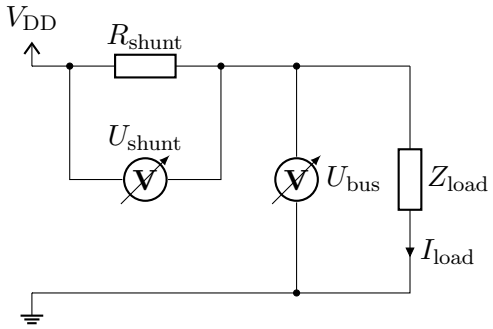

Figure 4: Power measurement using a shunt resistor

Since the voltage drop across the shunt is proportional to its resistance, that value should be chosen according to the expected current. It has to be large enough to cause a voltage drop that is easily measurable but as small as possible since this also consumes power and could significantly drop the supply voltage at high loads if chosen too large. The power consumed by the load can then be calculated as follows:

$$I_{\mathrm{load}} = \frac{U_{\mathrm{shunt}}}{R_{\mathrm{shunt}}} \tag{1}$$

$$P_{\text{load}} = U_{\text{bus}} \cdot I_{\text{load}} \tag{2}$$

The INA219 provides both voltmeters in fig. 4 and can also perform these calculations once it has been configured accordingly.

3.3.3 Voltage and Current References

Another important function of the ASICAB is to generate voltage and current references for the HICANN-X chip. The current reference is designed to improve temperature stability whereas the voltages are used during analog calibration. The *DAC6573* 4-Channel Digital to Analog Converter (DAC) (Texas Instruments, 2003) is used to generate the voltages, one of which is then converted into a current reference.

4 Software Development

As mentioned earlier, the software framework is a vital part of the BSS-2 neuromorphic system. To make the new hardware platform's power-regulation and -measurement functionalities accessible to the user, it has to be integrated into the hardware abstraction layers.

4.1 API Design

Many of the functionalities provided on the ASICAB are also available on the cube setup with varying degrees of compatibility. Sometimes the same functionality is provided using different hardware, sometimes there are subtle differences like the number of available channels. The goal was to be provide an API that is similar to the existing one for the cube setup while also addressing the hardware in a way that is natural to its mode of operation in order to keep complexity to a minimum.
Since all of the new peripherals are accessed via I2C (a very common communication standard for integrated circuits), introducing an abstraction for I2C registers provided a significant reduction in duplicate code.

4.2 Containers

Access to the hardware is provided via *haldls containers* which offer a moderate level of abstraction. Enabling a supply voltage using these containers is demonstrated in listing 1. The other components can be configured and read out in a similar fashion.

```
PlaybackProgramBuilder builder;
TCA9554Config config;

auto channel_outputs = config.get_channel_output();
channel_outputs[TCA9554ChannelOnBoard::vdd12_digital] = true;
config.set_channel_output(channel_outputs);
```

```
builder.write(TCA9554ConfigOnBoard(), config);
```

Listing 1: Enabling a supply voltage on the ASICAB board

4.3 Setup Initialization

The HICANN-X needs to be initialized before program execution in order to ensure proper function and increase repeatability. This includes resetting the internal and external state and setting all parameters that potentially could have changed during previous executions to their default values. For the Ultra96 setup this involves enabling the supply voltages and setting the DACs to their default reference voltages.

5 External Energy Measurement

Using the containers written for the *INA219* power-measurement ICs it is easy to measure power consumption at certain point in time during execution of a program. However, this method is not well suited for continuous sampling, since every access to the hardware must be commanded individually and is executed from the same pipeline as the experiment itself. This would lead to slower experiment-execution and relatively sparse sampling of power consumption.
To combat this, an external I2C master will be used to sample the power consumption. A RaspberryPi 4B was chosen for this task because it provides plenty of general purpose inputs and outputs (GPIO), an I2C interface and a fast processor for running the control software.

5.1 Experiment Sequencing

This setup also conforms to an externally specified evaluation procedure, which makes use of the control signals in fig. 5 in order to measure power consumption during different phases of a predefined workload. In order to validate that the setup responds to these control signals correctly, the same Raspberry as for the power measurement will be used to emulate this evaluation procedure. This also allows for correct attribution of power samples to the different phases of execution.

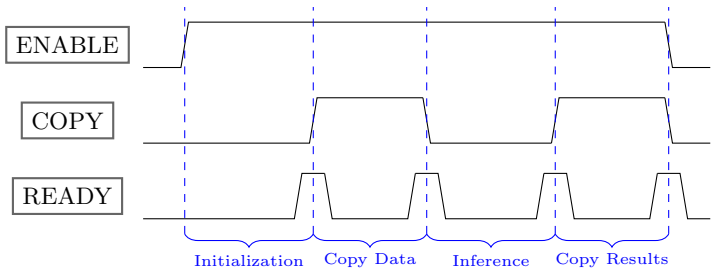

Figure 5: Evaluation sequence GPIO signals and resulting phases

5.2 Power Measurement

In order to observe quick variations in power consumption, high sample rates are necessary. Thus, the following optimizations were made.

1. ADC Resolution
   The INA219 supports different internal ADC acquisition-modes which offer a trade-off between resolution and conversion time. 12 bit resolution can be sampled every 532 µs whereas acquisition at 9 bit takes just 84 µs. Since 9 bit resolution is still adequate, this acquisition mode is used.

2. On-Chip Power Register
   As mentioned previously, the INA219 can also calculate the power consumption internally and exposes this value via a register, once configured to do so. This is desirable in this case, since only one register needs to be accessed per sample and channel which reduces bus traffic.

3. I2C Protocol Optimizations
   The I2C bus only has limited bandwidth and while we were able to achieve a clock-frequency of 1 MHz, the protocol should be as efficient as possible to maximize useful data throughput. One of these optimizations is the use of the INA's quick-read functionality, in which the register to be read is only sent once and all subsequent reads default to this register. We also make use of batched I2C transactions in the C++ code controlling the measurement to eliminate pauses between individual read transactions.

These optimizations result in a mean sampling period of $\sim 230\,\mu s$ ($\sim 4.35\,kHz$) where one sample consists of the power measurements of all seven channels.

5.2.1 IC Configuration

To make use of the internal power calculation, the INA219 needs to be configured with a calibration value. This calibration determines the power register's least significant bit (LSB) value, which is important for correctly interpreting the binary value. With the maximum expected current $I_{\max}$ and the shunt's resistance R_{shunt} the calibration can be calculated as follows.

$$\text{LSB}_I = \frac{I_{\max}}{2^{15}} \tag{3}$$

$$\text{CAL} = \text{truncate}\left(\frac{0.04096}{\text{LSB}_I \cdot R_{\text{shunt}}}\right) \tag{4}$$

Here *truncate* signifies rounding to the nearest integer towards zero. Once the value of CAL is saved into the IC's calibration register, it automatically computes the current and power registers on every update of its internal voltage registers. The power register is then scaled as follows.

$$\text{LSB}_P = 20 \cdot \text{LSB}_I \tag{5}$$

5.2.2 Ultra96 Internal Sensors

The Ultra96 development board consumes the majority of the total system power in this setup and knowledge of the consumption of individual components is very useful in optimizing the system's efficiency. Internally the board uses two five-channel power regulators to supply its components with their required voltages. These regulators are connected to an internal I2C bus that can be exposed on the ASICAB's external headers enabling acquisition of power-data in a similar fashion.

5.3 Energy Calculation

To calculate the energy consumption from a series of power-samples, a simple numerical integration approach is used. Assuming samples (t_i, p_i) where $i \in [i_i, i_f]$ for any given channel, the total power used in this sampling interval is calculated as follows,

$$E = \sum_{i=i_i}^{i_f-1} (t_{i+1} - t_i) \cdot \frac{p_i + p_{i+1}}{2} \tag{6}$$

which corresponds to linear interpolation in between samples.

6 Results

Using the techniques described above, good measurements of system's power consumption were achieved. As already mentioned in section 5.2, a mean sample period of about $\sim 230\,\mu s$ ($\sim 4.35\,kHz$) was achieved for the sensors on the ASICAB, whereas only $\sim 3.4\,ms$ ($\sim 294\,Hz$) could be achieved for the sensors on the development board. This is due to a larger number of channels, a less efficient protocol and a lower I2C frequency on that bus.

Channel	$min(P)$ [W]	$max(P)$ [W]	$mean(P)$ [W]	Energy [mJ]
HICANN-X Analog	0.246	0.257	0.251	42.2
HICANN-X Digital	0.437	0.447	0.444	74.7
HICANN-X Total	0.685	0.705	0.695	116.9
Ultra96 FPGA	0.563	0.563	0.563	94.7
Ultra96 CPU	1.47	1.72	1.51	252.8
Ultra96 Total	2.22	2.53	2.29	384.2
Setup Total	4.97	5.27	5.05	849.8

Figure 6: Power consumption and integrated energies for different components during the inference phase (168.3 ms)

The setup's power consumption during the initialization and inference phases of a typical execution can be seen in fig. 7. The MPSoC's power consumption exhibits greater fluctuations than the HICANN-X's while also accounting for the bulk of the total energy.

In fig. 6, the power- and energy consumption is presented in more detail for a subset of channels.

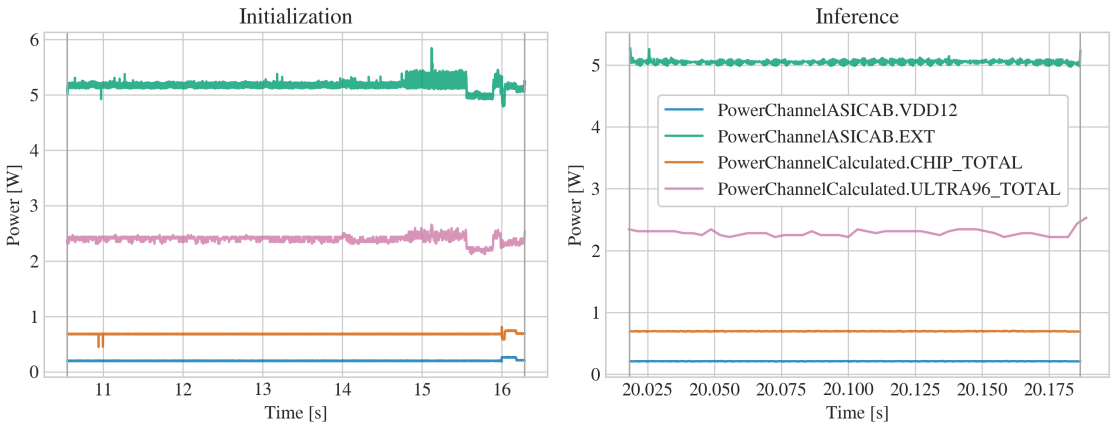

Figure 7: Power measurement during initialization and inference

7 Summary and Outlook

The Ultra96 Setup provides a portable, standalone single-chip setup and offers greatly improved energy efficiency over the older cube setups. The external controller enables detailed measurements of the power consumption and can also be used to validate that the control protocol is implemented correctly.
The API does not yet provide seamless compatibility with the new setup, since some components are not abstracted sufficiently. This could be addressed in the future by implementing higher-level containers that provide an interface compatible to that of the cube setup. As can be seen in the left graph of fig. 7, the high I2C bus frequency rarely causes bit-errors which leads to erroneous samples. This could be mitigated by using a lower bus frequency, as 1 MHz is far above the specifications of the I2C interface (see Raspberry Pi (Trading) Ltd, 2020, chap. 3.1). Another possibility would be to forego the external measurement completely and sample the power consumption asynchronously in the FPGA and return these samples in batches.

8 Acknowledgements

The work carried out in this report used systems, which received funding from the European Union's Horizon 2020 Framework Programme for Research and Innovation under the Specific Grant Agreements Nos. 720270, 785907 and 945539 (Human Brain Project, HBP), from the BMBF (16ES1127), and from the Lautenschläger-Forschungspreis 2018 for Karlheinz Meier.

References

Johannes Schemmel, Sebastian Billaudelle, Phillip Dauer, and Johannes Weis. Accelerated analog neuromorphic computing, 2020, 2003.11996.

Eric Müller, Christian Mauch, Philipp Spilger, Oliver Julien Breitwieser, Johann Klähn, David Stöckel, Timo Wunderlich, and Johannes Schemmel. Extending brainscales os for brainscales-2, 2020, 2003.13750.

Electronic Vision(s) Group. Coordinates for hicann-based and hicann-dls-based neuromorphic systems. `https://github.com/electronicvisions/halco`, a. [Online; accessed 20/01/2021].

Electronic Vision(s) Group. Fpga instruction set compiler for hicann. `https://github.com/electronicvisions/fisch`, b. [Online; accessed 20/01/2021].

Electronic Vision(s) Group. Hardware abstraction layer (and stateful encapsulation) for the hicann-dls. `https://github.com/electronicvisions/haldls`, c. [Online; accessed 20/01/2021].

Avnet Inc. Ultra96-v2 - xilinx zynq mpsoc development board. `https://www.avnet.com/wps/portal/us/products/avnet-boards/avnet-board-families/ultra96-v2/`, 2020. [Online; accessed 14/01/2021].

Texas Instruments. Ina219 zerø-drift, bidirectional current/power monitor with i2c interface. `https://www.ti.com/lit/ds/symlink/ina219.pdf`, 2015. [Online; accessed 14/01/2021].

Texas Instruments. Dac6573 quad, 10-bit, buffered voltage output dac with i2c interface. `https://www.ti.com/lit/ds/symlink/dac6573.pdf`, 2003. [Online; accessed 15/01/2021].

Raspberry Pi (Trading) Ltd. Bcm2711 arm peripherals. `https://datasheets.raspberrypi.org/bcm2711/bcm2711-peripherals.pdf`, 2020. [Online; accessed 20/01/2021].