Cavity in the Loop

David Scheunert
Computer Systems Group
Technische Universität Darmstadt
Darmstadt, Germany
scheunert@rs.tu-darmstadt.de

Jonas Gehrunger
Computer Systems Group
Technische Universität Darmstadt
Darmstadt, Germany
gehrunger@rs.tu-darmstadt.de

Christian Hochberger
Computer Systems Group
Technische Universität Darmstadt
Darmstadt, Germany
hochberger@rs.tu-darmstadt.de

Dieter Lens
GSI Helmholtzzentrum für Schwerionenforschung GmbH
Darmstadt, Germany
D.E.M.Lens@gsi.de

Harald Klingbeil*
Accelerator Technology Group
Technische Universität Darmstadt
Darmstadt, Germany
klingbeil@temf.tu-darmstadt.de

***Abstract*—Particle accelerators are physically and technically complex systems. Their operation requires demanding beam control systems. To support the development of new components in a beam phase control system for a synchrotron, we present a CGRA-based hardware/software environment that is capable of simulating the fundamental beam behaviour, leading to a hardware-in-the-loop setup. We show that this setup is able to emulate the longitudinal phase oscillations of the particle bunches in real-time.**

***Index Terms*—CGRA, real-time simulation, synchrotron LLRF system, closed-loop control**

I. Introduction

Particle accelerators and related experiments count among the largest technical facilities on earth. In ring accelerators like the synchrotron [1], [2], particles with an electric charge such as electrons or ions can be accelerated to high energies during thousands of revolutions. After acceleration, they can collide with other high-energy particles or with fixed targets. In a synchrotron, the particles do not move individually but in groups. They can either form a coasting beam, or they can be confined to so-called bunches if a radio frequency (RF) voltage is applied by means of one or more RF cavities. Multiple bunches can travel around the ring at the same time (then obviously with some time/position difference) if the frequency $f_{\mathrm{RF}} = h\, f_{\mathrm{R}}$ of the RF voltage is an integer multiple of the revolution frequency f_R. The factor h is called the harmonic number. Observing such a bunch leads to a pickup signal pulse which is often Gaussian but can have different distributions as well. All the bunches circulating in the synchrotron form the so called beam. In this paper, the discussed use cases refer to the GSI heavy ion synchrotron SIS18 in Darmstadt, Germany. There, bunches circulate in the synchrotron with a maximum revolution frequency of $f_{\mathrm{R}} \approx 1.4\,\mathrm{MHz}$ which translates to a minimum revolution time of $T_{\mathrm{R}} \approx 0.7\,\mu\mathrm{s}$.

A central part in such accelerators is the RF and beam control electronics, which is usually denoted as a Low-Level RF (LLRF) system [3]. It controls all beam manipulations (such as the acceleration, but also sophisticated bunch “gymnastics” [4]) in order to reach a high beam quality at the desired energy. The LLRF system makes use of the RF cavities in the synchrotron. It makes sure that the RF voltage along a so-called ceramic gap in the evacuated beam pipe (both are parts of the cavity) is stable and matches the requirements with respect to amplitude, frequency and phase.

Normally, a sinusoidal gap voltage is applied that acts on the bunches. In the longitudinal beam dynamics theory [5]–[7], one defines a reference particle that always has the correct parameters for the desired acceleration scenario. If a real particle arrives late ($\Delta t > 0$) at the gap, it experiences a higher voltage than the reference particle and is therefore accelerated. If the real particle is early ($\Delta t < 0$), it experiences a lower voltage and is slowed down. This behaviour is shown in Fig. 1. It leads to a complex nonlinear dynamical behaviour that can be visualised in longitudinal phase space where the energy of each particle is plotted over its position in time. In the longitudinal phase space, a bunch is represented by a huge number of individual particles.

The most simple case is that of a bunch circulating in the synchrotron at a constant energy. In this so-called stationary case, the bunch centre is located in the zero crossings of the sinusoidal gap voltage. In case there are undesired deviations from the desired ideal signals, it can happen that bunches oscillate as a whole around the zero crossing point. Such a longitudinal dipole oscillation is only one possible mode of oscillation. Others also exist, but are not discussed in this paper.

As stated above, the beam quality should be preserved by the LLRF system. Thus, it must be able to damp such longitudinal oscillations. For this purpose, the LLRF system includes a so-called beam phase control loop, which measures the longitudinal position of the bunches and actively changes the phase of the gap voltage in the cavities [8], [9]. It should be noted that at GSI, the voltages provided by different cavities can be several $10\,\mathrm{kV}$. The design and implementation of the electronic system to measure the beam and to produce the

*Also with GSI Helmholtzzentrum für Schwerionenforschung GmbH

979-8-3503-5554-3/24/$31.00 ©2024 IEEE
DOI 10.1109/SCW63240.2024.00272

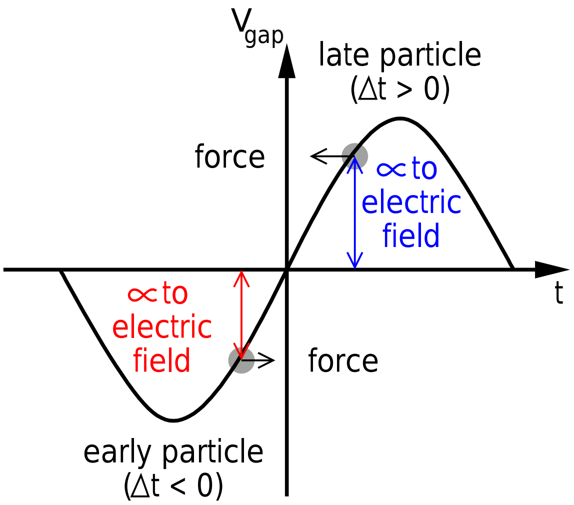


Fig. 1: Sample forces that influence a bunch

required signals for the cavities is extremely demanding.

Unfortunately, running such accelerators is a very cost intensive endeavour. Thus, very little beam time is devoted to the development of new components for the LLRF system or other accelerator components. This situation was the starting point for the research that is described in this paper. Instead of operating the beam phase control system in the real ring, we let it run against a simulation of the beam. This simulation can be used to test new components without interrupting the normal operation of the accelerator.

In order to produce meaningful results, the simulation must fulfil multiple requirements:

- The underlying beam model must resemble the true behaviour of the bunches close enough.
- The simulation must be fast enough to simulate bunches even at the highest circulation speed in real time.
- The simulation must interface to the beam control system, ideally without the need to change its setup.
- The time jitter of the outputs of the simulation must be very small compared to the time of circulation.

Since the bunch simulation code is implemented in a separate system, whereas the beam control system consists of the same hardware as used in the synchrotron, we arrive at a hardware-in-the-loop test bench setup. Such a hardware-in-the-loop setup only works if the simulation operates in hard real-time. The remaining paper primarily discusses the system to simulate the bunch behaviour.

After several investigations, we decided that a pure software based solution for the evaluation of bunch models is not feasible. In principle it could be fast enough, but the time jitter induced by the microarchitecture and the interfacing to the sensors was too high. An alternative could be a Field Programmable Gate Array (FPGA) implementation of the model. This would simplify the interfacing to the beam phase control system as the FPGA can be easily connected to Analogue/Digital and Digital/Analogue converters. Yet, it would make the development of the simulation very tedious, as we can expect hardware synthesis times of multiple hours. Thus, we ultimately decided to use an overlay architecture for the FPGA, which is called a Coarse Grained Reconfigurable Architecture (CGRA). It has the benefit of very fast programmability and at the same time its input/output timing can be controlled very precisely.

In this paper, we describe the setup, architecture, and programming of our CGRA based simulator. We evaluate the quality of our hardware-in-the-loop setup by comparing its output to the real beam behaviour. We will show that the same effects can be seen in both cases and the longitudinal oscillation matches very well between both cases.

The remaining paper is structured as follows. Section II discusses related research work in the relevant fields. It is followed by a description of our simulation system realised in an FPGA in Section III. In Section IV we describe the model that is used to simulate the bunch behaviour and how we have implemented it. This is followed by an experimental evaluation in Section V, where we compare the simulated behaviour to the real behaviour of the beam. Ultimately, we give a conclusion and an outlook onto future work in Section VI.

II. Related Work

CGRAs are well established in the literature [10]. They provide a trade-off between the flexibility that FPGAs can offer and the lower design complexity and faster iteration times of software. CGRAs offer high compute performance while allowing predictable real-time operation, making them suitable to be used in real-time control engineering environments, e.g. as was done in Ultrasynth [11]. Similarly, CGRAs have been found to be suitable for compute intensive simulation tasks [12] and even for low-power signal processing in wearable applications [13].

Particle tracking algorithms are widely used in the accelerator community. Tools like ESME [14], Long1D [15] or the BLonD code [16] are used for offline-simulation of the beam dynamics behaviour. They include many important beam dynamics effects that often have to be taken into account in realistic accelerator scenarios, such as beam loading or space-charge effects. Even on powerful computers, the computation time is of course far from the real-time requirements that stem from a hardware-in-the-loop setup.

The high performance of FPGAs has also been used in particle accelerators for years – not only for control loop applications with lower complexity, but also for example to implement real-time feedback control loops based on Reinforcement Learning [17].

III. Hardware Setup

A. System-side integration

The simulator introduced in this contribution is based on the AMD/Xilinx Virtex 7 VC707 FPGA [18] evaluation board. This board is extended with an FMC151 [19] daughter card consisting of a two-channel 14-bit Analogue Digital Converter

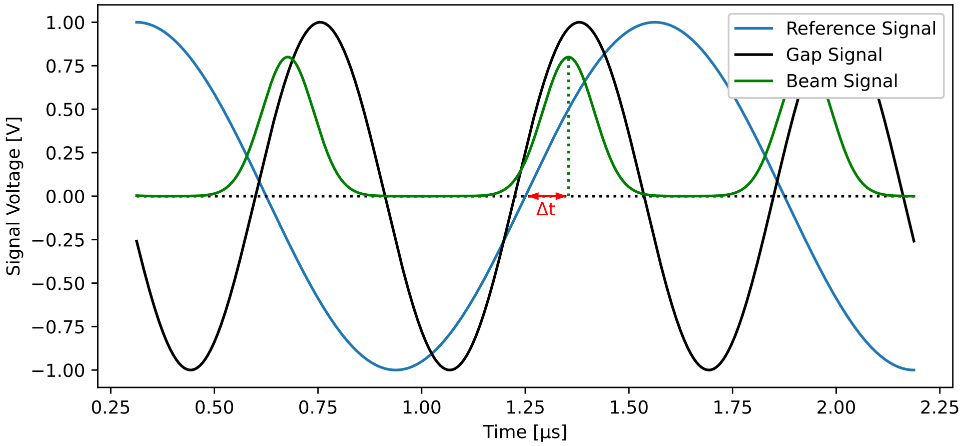


Fig. 2: Example for input and output signals with harmonic number h = 2 (non-equilibrium snap-shot)

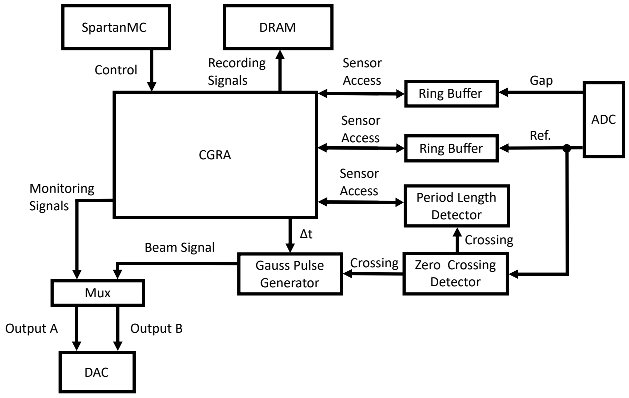


Fig. 3: Block diagram of the FPGA Framework design

(ADC) and a two-channel 16-bit Digital Analogue Converter (DAC), both running at 250 MHz sampling rate, with input and output amplitudes limited to $2V_{p-p}$ in the experiments. The interface with the beam phase control loop is using two input voltages, a reference signal and a gap signal, as shown in Fig. 2 in blue and black respectively. Output of the simulation is a beam signal consisting of Gaussian distributed pulses, shown in green colour. A second output channel is providing a monitoring signal to either show the phase difference calculated in the model or mirror the generated signal, this can be adjusted at runtime. Gap and reference voltage are scaled down on the beam side of the setup to fit within the acceptable ADC and DAC voltage ranges.

B. *FPGA Framework Design*

Inside the FPGA, a top level framework wraps all other simulator components. The architecture is shown in Fig. 3. The SpartanMC softcore processor [20] is a custom 18-bit processor optimised for FPGA architectures and serves as a parameter interface. It can control basic parameters of the simulation, adjust the scaling of output voltages and change which monitoring signal is produced. Furthermore, it allows to record the simulation into the DRAM memory of the FPGA board, which can be read out from a computer via the serial port.

Sample acquisition is done through a ring buffer for each input signal that needs to hold at least two full cycles of the reference voltage to accommodate for positive and negative Δt values. The voltages, as measured by the ADC, are recorded with the full 250 MHz sampling rate. A second port on each buffer allows the simulator to access a sample value in each cycle without interrupting the capturing process. For smaller revolution frequencies down to 100 kHz, up to 2500 samples have to be saved per revolution. To accommodate the requirement to hold two full cycles in the buffer, the buffers thus have the capacity to store $2^{13} = 8192$ input values.

One ADC channel provides the reference voltage input, which is also connected to a zero crossing detector. This module both measures the frequency and time of the last positive zero crossing of the sinusoidal input voltage. A period length detector determines the frequency of the reference signal. The measured frequency is averaged over the past four periods to reduce jitter. The simulation model is executed within a CGRA as described in section III-C. Output of the CGRA is the calculated Δt. Using the previous positive zero crossing and the current frequency, the correct time to trigger the next output Gauss pulse is stored in the Gauss pulse generator module. For the simulation covered by this contribution, a single module is used. When the timer module triggers, a single, precalculated, Gaussian distributed pulse is played back from sample memory through the DAC output.

C. *CGRA*

Model calculations are performed through a CGRA. CGRAs are a type of overlay architecture that consist of Processing Elements (PEs), where each PE can have its own set of operators to perform numerical operations, with a selection ranging from pure integer arithmetic to floating point operations up to `Coordinate Rotation Digital Computer` (CORDIC) for trigonometric functions. For this experiment, basic floating point and square-root operators are in use. Each PE is connected to its surrounding neighbours through a configurable interconnect. Results of operations can be passed on, allowing the routing of operands where no direct connection exists. The framework design covered in section III-B is agnostic to the CGRA configuration, allowing an arbitrary number of PEs (e.g. 3x3 or 5x5) and any interconnect structure without manual changes to the surrounding implementation.

To connect the CGRA to the simulator, a `SensorAccess` module was implemented to act as memory. This allows the simulation model to both read input signal data and set the output timing for the next Gauss pulse.

The same CGRA framework has been used in a another hardware-in-the-loop setup in [11]. In this setup, the oscillations of a motor vehicle during a test drive were captured. In order to find the sources of noise in the motor vehicle design, the test setup could then reproduce the exact vehicle motions of this test drive as many times as needed in a static test environment. This demonstrates the flexibility of our CGRA framework.

Programming of the CGRA is done using the `C` programming language. A code parser converts the program into a `Scheduler Application Representation` (SCAR) control and data flow graph format, which is processed by the CGRA scheduler. The scheduler is a customised resource-constrained list scheduler [21]. Output of the scheduler are the contents for all context memories, which can be inserted into the final FPGA bitstream without requiring a new synthesis. This allows very fast iterations of the model, as changes to the C implementation are available on the experimental setup in seconds.

IV. Model

In the following section we will give a detailed explanation of the model we have implemented in order to simulate the behaviour of the ion beam.

A. Particle Tracking Equations

The particles in the synchrotron accelerator reach velocities well over ninety percent of the speed of light, so the calculations have to be done in terms of special relativity. The nomenclature used in this paper follows the definitions given in [22] and will be explained briefly. Two so-called Lorentz factors are used to describe the current energy of the ions:

$$\beta_v = \frac{v}{c}; \quad \gamma_v = \frac{1}{\sqrt{1-\beta_v^2}}. \tag{1}$$

The first factor, β_v, refers to the fraction of the speed of light c that the particle velocity has reached. The second factor, γ_v, describes the factor by which the energy of the ion has increased in comparison with the energy at rest due to special relativity. These factors are interdependent, so knowing one of them is sufficient for all further calculations.

In order to simulate longitudinal oscillations our model of the ion beam consists of only two particles. One so-called reference particle (index R), which will always remain on the same orbit resulting in a constant orbit length l_R and an asynchronous particle, which can oscillate around the reference particle. Quantities that apply for the reference particle are marked by the index R whereas quantities that are valid for the asynchronous particle have no such index R. For example, the momentum of the reference particle in the n-th revolution is denoted as $p_{R,n}$ whereas the momentum of the asynchronous particle in the n-th revolution is p_n. Another example is the relativistic gamma so that $\gamma_{R,n}$ refers to the reference particle and γ_n to the non-synchronous particle (both in the n-th revolution). Since the orbit length l_R of the reference particle is the same in each revolution, it has no index n, whereas the orbit length l_n of the asynchronous particle depends on the revolution.

Under normal circumstances, a huge number of asynchronous particles must be assumed to represent a realistic bunch, but in our case we model a whole bunch by one asynchronous macro particle. The reference particle does not exist in the physical world, but rather is a mathematical construct to describe the longitudinal oscillation. The energy of the reference particle at a given revolution n can be calculated recursively by adding the energy gain of the particle in the electric field at the gap[1] to its energy of the previous revolution. Due to the high velocity of the particle, it can be assumed that the gap voltage $V_{R,n-1}$ is constant during the time the particle is exposed to the electric field. This leads to the following equation which recursively determines the Lorentz factor $\gamma_{R,n}$:

$$\gamma_{R,n} = \gamma_{R,n-1} + \frac{Q}{mc^2} V_{R,n-1} \tag{2}$$

Here, Q is the charge of each particle, and m is its rest mass.

Furthermore, quantities for the asynchronous particle are expressed as differences from the quantities of the reference particle using a Δ-symbol.

The difference of the Lorentz factor can be obtained using

$$\gamma_n - \gamma_{R,n} = \Delta\gamma_n = \Delta\gamma_{n-1} + \frac{Q}{mc^2} \Delta V_{n-1}. \tag{3}$$

Since the energy of the asynchronous particle differs from the energy of the reference particle, and because the magnetic field is controlled in a way to keep the reference particle in a constant orbit, the asynchronous particle will not move on the reference orbit. The momentum compaction factor α_c can be used to relate a change in the particle momentum to a change in orbit length according to (4). α_c is a factor specific to each

[1] The energy gain of the particles takes place inside a so-called gap which is an interruption of the metallic beam pipe by a short ceramic section. Such a ceramic gap inside a cavity allows the induction of a radio frequency voltage that accelerates the charged particles passing the gap.

individual particle accelerator and the chosen ion optics. For the accelerator at GSI, as in most cases, α_c is a positive value.

$$\frac{\Delta l_n}{l_R} = \alpha_c \frac{\Delta p_n}{p_{R,n}} \tag{4}$$

Following the scheme explained above, p_n is the momentum of the asynchronous particle in the n-th revolution whereas $p_{R,n}$ is the momentum of the reference particle in the n-the revolution. The delta quantity Δp_n is the difference of these two quantities.

The momentum compaction factor can be used to compute the phase slip factor η_R. This factor has a meaning similar to α_c but instead of relating the change in orbit length to the change in momentum, it relates the change of the revolution time to the change in momentum.

$$\frac{\Delta T_n}{T_{R,n}} = \eta_{R,n} \frac{\Delta p_n}{p_{R,n}} \quad \text{with} \quad \eta_{R,n} = \alpha_c - \frac{1}{\gamma_{R,n}^2} \tag{5}$$

By means of the following three simplifications of the model, equation (6) can be derived, which is a recursive way of calculating the difference in the arrival time[2] of the asynchronous particle with respect to the reference particle.

$$\frac{\Delta\gamma_n}{\gamma_{R,n}} \ll 1 \ ; \ \frac{\Delta\beta_n}{\beta_{R,n}} \ll 1 \text{ and } \frac{\Delta\beta_n}{\beta_{R,n}} \approx \frac{1}{\gamma_{R,n}^2 \beta_{R,n}^2} \frac{\Delta\gamma_n}{\gamma_{R,n}}$$

$$\Delta t_n = \Delta t_{n-1} + \frac{l_R \eta_{R,n}}{\beta_n \beta_{R,n}^2 c} \frac{\Delta\gamma_n}{\gamma_{R,n}} \tag{6}$$

The iterative calculation of equations (2) and (6) can now be used to update the phase space position of the asynchronous particle from revolution to revolution (so-called particle tracking). This concludes the brief overview of the mathematical foundations of the longitudinal phase space model for the synchrotron. A more in-depth explanation can be found in [22].

B. C Implementation

In this section we present the implementation details of our simulation and outline the steps taken to translate the mathematical model shown in the previous section into functional code. The model is implemented in C and compiled for our CGRA hardware.

The reference signal is delivered by a so-called Group Direct Digital Synthesis (DDS) module. It generates a sine wave that follows the revolution frequency set values in an undisturbed way. For the stationary case, its positive zero crossing is regarded as the position of the reference particle. The phase of the gap signal is actively controlled by the beam phase control system depending on the beam signal, which is measured by a pickup sensor in reality but generated by our simulator to realise the cavity in the loop emulation.

Due to the recursive nature of (2), (3) and (6) it is essential that the coding provides a proper initialisation. Therefore, the program first waits for a valid measurement of four full sine waves before starting the initialisation process. In this initialisation step, the program requests the revolution time of the reference particle measured by the period length detector. The sensor returns the number of clock cycles between two positive zero crossings of the reference sine wave. To increase accuracy, the sensor applies a simple average filter by accumulating the last four period lengths measured. Knowing the length of the orbit and the sampling frequency of our sensor, we can compute the velocity of the reference particle. Using (1) we can obtain initial values for $\beta_{R,0}$ and thus for $\gamma_{R,0}$.

The input voltages of course do not contain any information about the asynchronous particle. Therefore, initial values for $\Delta\gamma_0$ and Δt_0 must be provided as constants in the code. In our test case, we do not want to stimulate an oscillation by hardcoding the initial values, but by changing the input voltages and then damping the oscillation again by means of closed-loop control. Therefore we set these initial values to zero.

Once the initialisation is complete, we enter the main loop, which we will go through once per particle revolution. The main loop starts by requesting the most recent revolution time of the sine wave from the period length detector. Using the Lorentz factor from the previous iteration, we can calculate the time it took the reference particle to complete one revolution at the energy it currently has. By calculating the difference between these two revolution times ($\Delta T_{R,n-1}$) we know the time relative to the zero crossing of the reference signal at which the reference particle reached the gap. This relative time is translated into a number of clock cycles via the sampling frequency and then sent as an address to the ring buffer capturing the reference signal. The buffer, using the address of the zero crossing as address offset, returns the voltage measured for that time step. Since $\Delta T_{R,n-1}$ is rarely ever an integer multiple of the period length of the sampling frequency, a second value is requested from the buffer to perform linear interpolation to increase the accuracy. This value is then multiplied by a scaling factor to obtain $V_{R,n-1}$, since the voltage at the input of the ADC is several orders of magnitude smaller than the voltage at the gap.

A similar procedure is performed for the asynchronous particle to obtain the voltage V_{n-1}. In this case, the different revolution time of the asynchronous particle needs to be taken into account by adding Δt_{n-1} to $\Delta T_{R,n-1}$ before requesting the voltage from the ring buffer that captures the gap signal.

With the voltage $V_{R,n-1}$ we can update the Lorentz factors for the reference particle according to (2) and with this a new value for the phase slip factor can be found using (5). The difference between $V_{R,n-1}$ and V_{n-1} inserted in (3) allows us to compute the updated $\Delta\gamma_n$. Using the Lorentz factor $\gamma_{R,n}$ this also allows us to calculate γ_n. Inserting all our results so far into (6), we obtain the updated value Δt_n. Finally, we write back the Δt_n to an actuator responsible for generating the output of our simulator, as this is the time relative to the zero crossing of the reference signal voltage when the simulated bunch reached the gap.

Due to the strict real-time requirements of the system, i.e. that the calculation must be completed within one period

[2] By arrival time we mean the time when the particle arrives at the cavity gap.

length of the reference sine wave, which can be faster than one microsecond, it is not possible to simply use the C-model as described above. It is too sequential and therefore too slow, because it does not take full advantage of the parallel structure of our CGRA. To reduce the sequential nature, we manually pipelined the loop by a factor of two. This means that the execution of a single calculation takes two iterations of the loop, however, there is still a new calculation started and finished each iteration. At the end of the loop any results from the first loop iteration that are needed for the second iteration are assigned to new variables. This allows the scheduler to run both parts in parallel, as they do not depend on the results they produce in this iteration, but on the results of the previous iteration instead.

The pipelining is done by splitting the loop after the voltages have been calculated. In addition, the write back operation for Δt_n is pushed into the first loop iteration as it is already known. Therefore, all IO operations are performed in the first loop iteration which means that there is no additional delay induced by the loop pipelining.

With this pipelining, we achieve a schedule length of 111 clock ticks while calculating 8 bunches per revolution. This is significantly faster than the calculation without loop pipelining which results in a schedule length of 128 clock ticks. In order to meet timing criteria on our FPGA, we cannot use the system clock of $250\,\mathrm{MHz}$ for our CGRA. Therefore, the CGRA uses its own clock running at $111\,\mathrm{MHz}$. This means, we can simulate particles with revolution frequencies of up to $1\,\mathrm{MHz}$ due to our loop pipelining instead of the $\approx\ 867\,\mathrm{kHz}$ without loop pipelining. We can further reduce this schedule length by reducing the number of bunches simulated. By simulating only four bunches, we shrink down the length of our schedule to a total of 99 clock ticks. And if only a single bunch is simulated, the schedule length is even further reduced to 93 clock ticks. Doing so, alows us to simulate particles with revolution frequencies of $\approx\ 1.12\,\mathrm{MHz}$ or with revolution frequencies of up to $\approx\ 1.19\,\mathrm{MHz}$ respectively.

V. Evaluation

In the following section we present an assessment of the quality of our simulator based on the results of an experimental test. The aim of the evaluation is to demonstrate the feasibility of a hardware-in-the-loop system that can accurately emulate a simplified model of the beam and thus demonstrate the effectiveness of our simulator, which lies not only in its real-time capability, but also in its fast reconfiguration time measured in seconds.

In order to properly evaluate the quality of our simulator, we used the same parameters for our setup that were used in a machine development experiment (MDE) at the GSI heavy-ion synchrotron SIS18 on Nov. 24th 2023. During this MDE, longitudinal oscillations of the bunches have been excited deliberately by generating periodic phase jumps, and a closed-loop control electronics system was used to damp these oscillations.

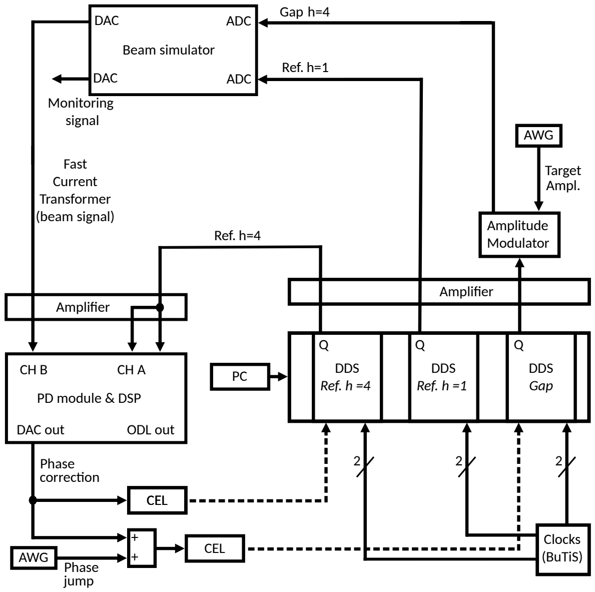
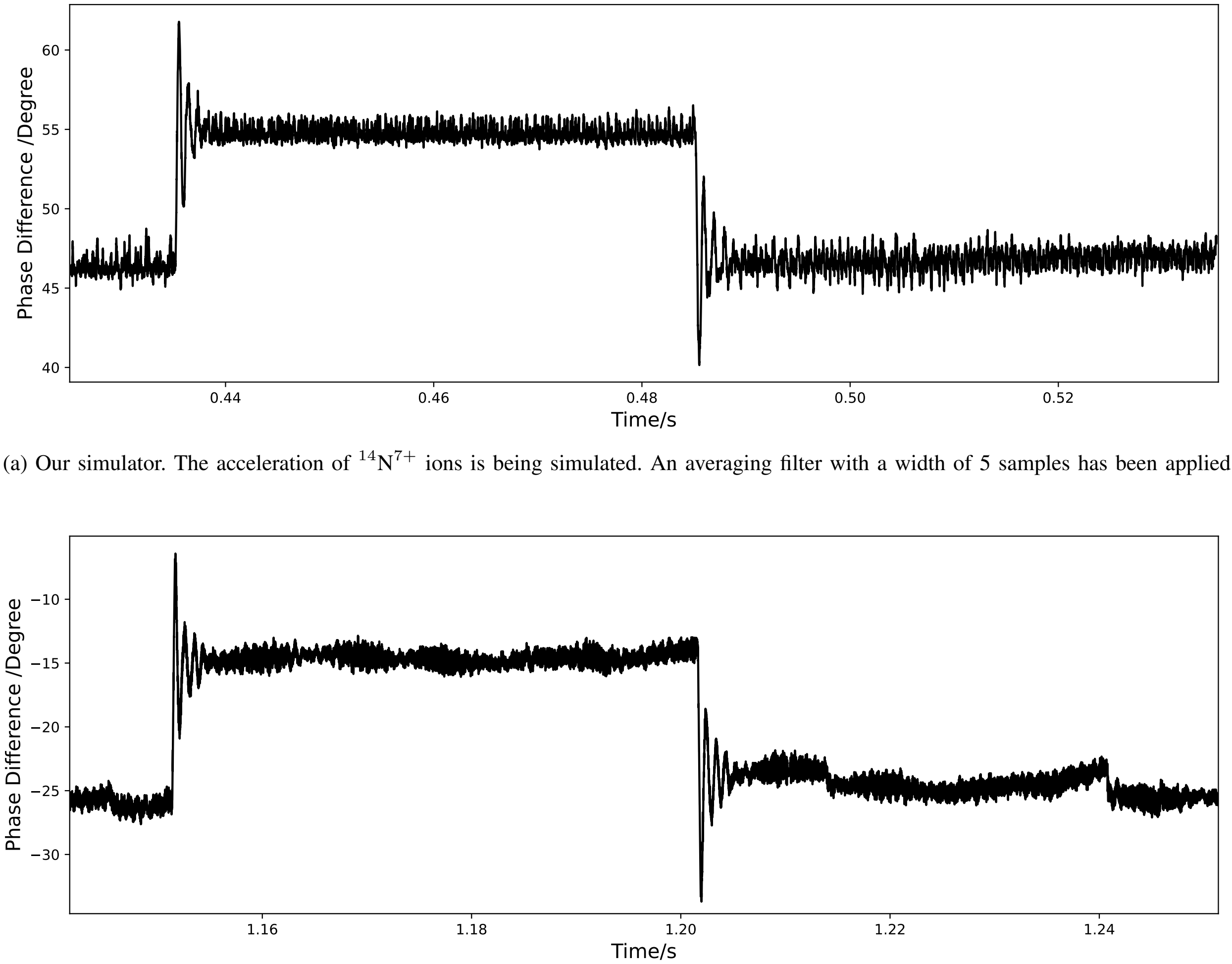


Fig. 4: Experimental setup

The structure of our experimental setup is shown in Fig. 4. Three DDS modules were used to generate three synchronised RF signals that have the desired phase relationship [23]. The synchronicity is ensured by simultaneously resetting the phase of all DDS modules using a mini control system and by using the synchronised clock signal from the BuTiS (Bunch Phase Timing System) of GSI. This system is used to provide a low-jitter (in the low femtosecond range) synchronous clock signal throughout the accelerator facility campus with an accuracy of 100 picoseconds per kilometre [24]. The gap voltage must be phase-shifted with respect to the reference voltage. In our test setup, this phase jump is created as an analogue signal via an arbitrary waveform generator (AWG) and converted into an optical stream via a Calibration Electronics (CEL) module. The optical stream is then fed back to the DDS. The phase jump was toggled every twentieth of a second. The constructed signal that emulates the gap signal of a cavity contains 8 degree jumps which is only slightly different from the 10 degree jumps made in the machine experiment.

Analogue to the setup in the operational accelerator, a DSP system is used to capture the phase difference between the beam signal generated by our simulator and the reference signal. The measurements were performed with a harmonic number of four, which means that there are four bunches in the accelerator and therefore the frequency of the gap voltage is four times the frequency of the reference voltage. Similar to the MDE, we used input frequencies of $800\,\mathrm{kHz}$ for the reference signal and $3200\,\mathrm{kHz}$ for the gap signal respectively.

The frequency of the longitudinal oscillations, the synchrotron frequency, was measured to be $1.2\,\mathrm{kHz}$ in the MDE. Therefore, for our measurement, the input voltage amplitude was adjusted to achieve a similar synchrotron frequency of



(a) Our simulator. The acceleration of $^{14}N^{7+}$ ions is being simulated. An averaging filter with a width of 5 samples has been applied.

(b) GSI heavy-ion synchrotron SIS18 on Nov 24th, 2023. $^{14}N^{7+}$ ions are being accelerated.

Fig. 5: Measurement of the difference in phase between the reference signal and the beam signal.

$1.28\,\text{kHz}$. The closed-loop control system uses a Finite Impulse Response (FIR) filter. The parameters of the closed-loop control were set to $f_{pass} = 1.4\,\text{kHz}$, gain $= -5$ and recursion factor $= 0.99$, which are the optimal parameters according to [8].

Thus, our test setup realistically emulates the MDE setup.

The aim of the experiment with our real-time simulator was to show, that our system reacts in a similar way to the SIS18, i.e. that the phase jump would cause the simulated bunches to oscillate at a realistic amplitude and frequency, and that the closed-loop control would manage to damp the oscillation as in the mentioned MDE. As you can see in Fig. 5, our measured results match the expectations.

The results show a constant offset in the phase difference because the dead times in the test setup differ from those in the real sychrotron (due to the position of the synchrotron components and due to different cable lengths). However, this offset is not important for the evaluation of our experiment because only the relative differences of the phase before and after the jump matter. There is also a constant offset between the two time axes, which is also irrelevant since the time interval between the phase jumps is identical. Apart from these two minor differences, there is a remarkable similarity between the behaviour of the simulator and real accelerator.

The beam signal of the simulator realistically reacts to a phase jump by starting to oscillate. Initially, the peak-to-peak phase amplitude of this oscillation is twice the amplitude of the phase jump. This can be observed on the first peak after each jump and is in line with the expected behaviour which can also be observed in the SIS18 experiment.

The control loop is effective in damping the longitudinal dipole oscillation. A strong decrease of the amplitude of the phase oscillation can be observed. Without the control loop, the real particle bunch in the accelerator would also experience a decrease of the phase oscillation amplitude due to Landau damping [25] and filamentation. In this context, the damping of the coherent oscillation results from the large number of particles in the bunch which have a certain energy spread. Although each particles oscillates with its synchrotron frequency, the oscillation of the bunch centre will therefore decrease. This effect is obviously not simulated in our system, as we have simplified our model to include only a single macroparticle. It would require the simulation of tens of thousands of individual particles to see this effect. However, since the damping from the control loop is much stronger, the effect of filamentation and Landau damping can be neglected for the controlled system. Therefore, our results closely match those measured in the MDE, as can be seen in Fig. 5.

VI. Conclusion

In this paper we have demonstrated that a hardware-in-the-loop setup for a complex LLRF system in a synchrotron is possible. Its real-time performance allows that the beam control system can be run against this simulation. The simulated bunch behaviour shows similar longitudinal bunch oscillations as observed in machine development experiments with the real beam. The usage of a CGRA to carry out the simulation has proven extremely useful as the turn-around time after model changes is only in the range of seconds (compared to a full FPGA synthesis that can easily take hours).

Currently, we are also implementing the ramp-up case, which simulates the bunches after injection into the ring. At that point bunches have much smaller energies and longer revolution times. Therefore, the challenge is to emulate the acceleration phase with variable RF frequencies and amplitudes.

In the future, we want to improve the level of detail in our simulation by replacing the single macro particle with a set of macro particles. This will allow us to simulate other oscillation modes (like quadrupole oscillation). Also, it allows us to replace the synthetic Gauss pulse by a parametric version that adapts to the energy/phase distribution of the bunch. Ultimately, we will also extend the simulation to support multiple bunches circulating in the ring at the same time.

References

[1] A. W. Chao, M. Tigner, H. Weise, and F. Zimmermann, *Handbook of Accelerator Physics and Engineering; 3rd ed.* Singapore: World Scientific, 2006.

[2] E. J. N. Wilson, *An Introduction to Particle Accelerators*. Oxford Univ. Press, 2001.

[3] P. Baudrenghien, "Low-level RF - Part I: Longitudinal dynamics and beam-based loops in synchrotrons," 2011, CERN Accelerator School: Specialised Course on RF for Accelerators; 8 - 17 Jun 2010, Ebeltoft, Denmark.

[4] R. Garoby, "RF gymnastics in synchrotrons," 2011, CERN Accelerator School: Specialised Course on RF for Accelerators; 8 - 17 Jun 2010, Ebeltoft, Denmark. [Online]. Available: http://cds.cern.ch/record/1407406

[5] D. A. Edwards and M. J. Syphers, *An Introduction to the Physics of High Energy Accelerators*. WILEY-VCH Verlag GmbH & Co. KGaA, 2004.

[6] S.-Y. Lee, *Accelerator Physics*. World Scientific, 1999.

[7] P. J. Bryant and K. Johnsen, *The Principles of Circular Accelerators and Storage Rings*. Cambridge: Cambridge Univ. Press, 1993.

[8] H. Klingbeil, B. Zipfel, M. Kumm, and P. Moritz, "A Digital Beam-Phase Control System for Heavy-Ion Synchrotrons," *IEEE Transactions on Nuclear Science*, vol. 54, no. 6, pp. 2604–2610, 2007.

[9] J. Grieser, J. Adamy, T. Ferrand, K. Groß, U. Hartel, H. Klingbeil, U. Laier, D. Lens, K.-P. Ningel, S. Schäfer, and B. Zipfel, "A Digital Beam-Phase Control System for a Heavy-Ion Synchrotron With a Dual-Harmonic Cavity System," *IEEE Transactions on Nuclear Science*, vol. 61, no. 6, pp. 3584–3591, 2014.

[10] A. Podobas, K. Sano, and S. Matsuoka, "A Survey on Coarse-Grained Reconfigurable Architectures From a Performance Perspective," *IEEE Access*, vol. 8, pp. 146 719–146 743, 2020.

[11] D. Wolf, A. Engel, T. Ruschke, A. Koch, and C. Hochberger, "UltraSynth: Insights of a CGRA Integration into a Control Engineering Environment," *Journal of Signal Processing Systems*, vol. 93, no. 5, pp. 463–479, May 2021. [Online]. Available: https://doi.org/10.1007/s11265-021-01641-7

[12] A. Podobas, "Q2Logic: A Coarse-Grained FPGA Overlay targeting Schrödinger Quantum Circuit Simulations," in *2023 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2023, pp. 460–467.

[13] J. Lopes, D. Sousa, and J. C. Ferreira, "Evaluation of CGRA architecture for real-time processing of biological signals on wearable devices," in *2017 International Conference on ReConFigurable Computing and FPGAs (ReConFig)*, 2017, pp. 1–7.

[14] J. A. MacLachlan, "Particle tracking in E - ϕ space for synchrotron design and diagnosis," United States, Tech. Rep., Nov 1992, fNAL/C–92/333. [Online]. Available: http://inis.iaea.org/search/search.aspx?orig_q=RN:24038475

[15] S. R. Koscielniak, "The LONG1D simulation code," Canada, Tech. Rep., 1988, tRI-PP–88-41. [Online]. Available: http://inis.iaea.org/search/search.aspx?orig_q=RN:24059378

[16] H. Timko, J. Esteban Müller, A. Lasheen, and D. Quartullo, "Benchmarking the Beam Longitudinal Dynamics Code BLonD," p. WEPOY045, 2016.

[17] W. Wang, M. Caselle, T. Boltz, E. Blomley, M. Brosi, T. Dritschler, A. Ebersoldt, A. Kopmann, A. Santamaria Garcia, P. Schreiber, E. Bründermann, M. Weber, A.-S. Müller, and Y. Fang, "Accelerated Deep Reinforcement Learning for Fast Feedback of Beam Dynamics at KARA," *IEEE Transactions on Nuclear Science*, vol. 68, no. 8, pp. 1794–1800, 2021.

[18] Advanced Micro Devices, Inc., "AMD Virtex 7 FPGA VC707 Evaluation Kit," https://www.xilinx.com/products/boards-and-kits/ek-v7-vc707-g.html, [Accessed 16-08-2023].

[19] Abaco Systems, "FMC151 FPGA Mezzanine Card," https://www.abaco.com/products/fmc151-fpga-mezzanine-card, [Accessed 16-08-2023].

[20] G. Hempel and C. Hochberger, "A resource optimized Processor Core for FPGA based SoCs," in *10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2007)*, 2007, pp. 51–58.

[21] R. Wirsch and C. Hochberger, "Towards Transparent Dynamic Binary Translation from RISC-V to a CGRA," in *Architecture of Computing Systems*, C. Hochberger, L. Bauer, and T. Pionteck, Eds. Cham: Springer International Publishing, 2021, pp. 118–132.

[22] H. Klingbeil, U. Laier, and D. Lens, *Theoretical Foundations of synchrotron and Storage Ring RF Systems*. Springer Cham, 2015.

[23] A. Andreev, D. Lens, and H. Klingbeil, "A calibration system for hadron synchrotrons with a large frequency swing," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 1053, p. 168382, 2023.

[24] B. Zipfel, P. Moritz *et al.*, "Recent progress on the technical realization of the bunch phase timing system BuTiS," *Proceedings of IPAC*, 2011. [Online]. Available: https://accelconf.web.cern.ch/ipac2011/papers/mopc145.pdf

[25] H. G. Hereward, "Landau Damping," in *CAS - CERN Accelerator School: Advanced Accelerator Physics*, Oxford, September 1985, pp. 255–263.