DOI 10.5162/SMSI2025/A5.4

Ultrasonic phased array interface using programmable I/O and microprocessor clock synchronisation

Tom Nellius[1], Kevin Henne[1], Maximilian Hartinger[1], Lars Meihost[1], Tim Hetkämper[1], Henning Zeipert[1], Leander Claes[1], and Bernd Henning[1]
[1]Measurement Engineering Group, Paderborn University, Paderborn, Germany
meihost@emt.uni-paderborn.de

Summary: This contribution constitutes an approach to ultrasonic phased array excitation using programmable input/output banks for fast, digital signal generation, a method to synchronise a number of microprocessors and a minimalistic analogue frontend. Exploiting the low-pass characteristic of the array's transducers, it is assumed sufficient to apply a binary signal to each element. These signals are generated synchronously by state machines implemented in hardware, a feature present on some microcontrollers designated as programmable input/output. To increase the number of channels, the clock of a number of microcontrollers is synchronised using a phase detection circuit. A digital-input MOSFET low-side driver circuit is used for each channel to amplify the digital signals. Synchronous signal generation for a 64-element array is demonstrated using schlieren imaging.

Keywords: Clock synchronisation, Interface, Programmable I/O, Signal generator, Ultrasonic phased array

Motivation

Controlling ultrasonic phased array systems commonly requires complex, specialised, multichannel hardware. Especially when many elements are to be excited simultaneously, an equal number of amplifiers and signal generators is required, increasing hardware size and costs. For an efficient, robust implementation, it is thus advantageous to minimise the hardware required for each channel. To facilitate this, it is assumed sufficient to drive the elements of the array with binary signals, taking advantage of their low-pass characteristics. If the binary signal can be modulated with sufficient frequency, coded signals can be used to generate arbitrary signals after the low-pass filtering. This circumvents the necessity of a large number of digital-to-analogue converters. Binary signals can further be generated using general-purpose outputs of microcontrollers, which are available on nearly every pin of a microcontroller, compared to digital-to-analogue converters.

The aim is to drive a 64-element array with up to 5 MHz. Nevertheless, the critical parameter is not the signal frequency itself, but the time delay possible between the different transducer signals. This delay is needed to perform beam steering and/or focussing [1]. Thus, to have fine-grained control, it is desirable to have an output switching frequency as high as possible, ideally the system's clock. This does not apply to most microcontrollers. However, some microcontrollers (e.g. RP2040, *Raspberry Pi Foundation*, [2]) are equipped with hardware state machines, which can be connected to the output pins. This feature is called programmable input/output and enables independently switching a number of outputs with up to the system's clock frequency. Currently, available microcontrollers with programmable input/output do not provide a sufficient number of outputs to provide signals for 64 elements. It is thus necessary to realise an external clock synchronisation of a number of microcontrollers. Further, the output current of the microcontrollers is insufficient to drive the elements of the array. The behaviour of each element can be approximated by a capacitance, requiring a fast driver circuit for capacitive loads.

Implementation

The RP2040 microcontroller used to implement the phased array interface includes programmable input/output with eight parallel hardware state machines. Each state machine is fed by two first-in-first-out (FIFO) registers and can be run with frequencies up to the system clock (133 MHz) [2]. To output binary signals to the phased array, the desired binary sequences are generated by a personal computer, transferred to the microcontroller, and stored in the controllers RAM to be transferred into a FIFO-register on data request. A specialised instruction set is used to program these state machines, with each instruction requiring a single cycle to execute. The instruction set includes an instruction to pull values from the FIFO-registers and write them to the output pins:

```
.program fifo_to_out_pin

loop:
    pull
    out pins, 16
    jmp loop
```

The `pull` instruction transfers a 32 bit word from the FIFO-register to an intermediate register called output shift register (OSR). The `out` instruction writes the first 16 bit of the word in the OSR to 16 pins determined by the controller's configuration and the `jmp` instruction resets the program counter to the beginning of the `loop` block. The programmable input/output functionality can also be configured to perform this operation in a single instruction using a feature called *autopull* and by configuring the state machine to handle the loop automatically, rendering the *pull* and *jmp* instructions obsolete. Programmable input/output is also used to trigger the state machines to start

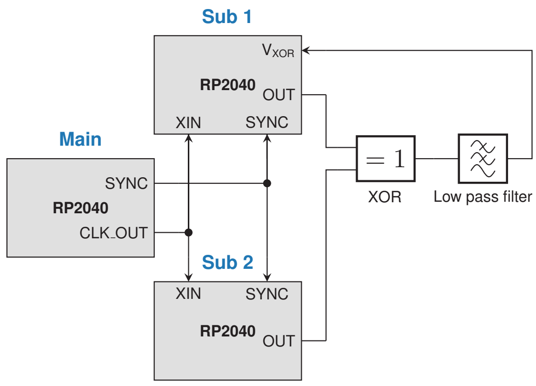

Fig. 1: Synchronisation setup with XOR phase detection between two microcontrollers.

outputting the signals. On the RP2040, 30 pins are available. With one pin used as a trigger input, 29 pins are available to output signals. For the presented application, the signals for 16 channels of the array are generated by a single controller.

To implement the required number of channels four microcontrollers are used, which need to operate synchronously. The microcontroller at hand provides a pin designated XIN, which is directly linked to the system's internal phase-locked loop (PLL). To validate the synchronicity of two XIN-synced controllers, a periodic signal (SYNC) is applied to all controllers and replicated using programmable input/output at the OUT pins, see Figure 1. If the two controllers are synchronised their generated output would be identical, but a comparison shows a phase-shift (up to $\pm 1/2$ clock period) between the generated signals. This phase-shift, possibly caused by the post-dividers of the PLL, is generated at random during the initialisation process. For the quantification of the phase-shift an XOR-gate with analogue low-pass filter at the output is utilised. The output of this circuit is at high-level if both input signals are complementary and low-level if they are identical. Hence, while observing this output of the phase-detection circuit with an analogue-to-digital converter at V_{XOR}, one controller's PLL is reset until the phase-shift is (close to) zero. On average, this synchronisation process is completed under 1 s. As this clock synchronisation only needs to be carried out when powering up the array interface, it is considered sufficiently fast.

The current at the output of the microcontrollers is insufficient to drive the elements of the array. A minimalistic driver circuit is developed based on a low-side gate driver integrated circuit (UCC27517A-Q1, *Texas Instruments*). These integrated circuits (IC) are designed to drive the gates of large MOS transistors with high frequency signals and are thus particularly suited to drive other capacitive loads, such as piezoelectric elements of a phased array. The specific IC used here also provides a digital input u_{in} that can be directly connected to the output pin of a microcontroller. A supply voltage U_0 of 18 V is used, which defines the high level output voltage and thus the voltage at the elements of the array Z_{load}. No further active or passive components are required, however it is advisable to buffer the supply voltage U_0 with sufficiently large capacitances (220 nF) for each

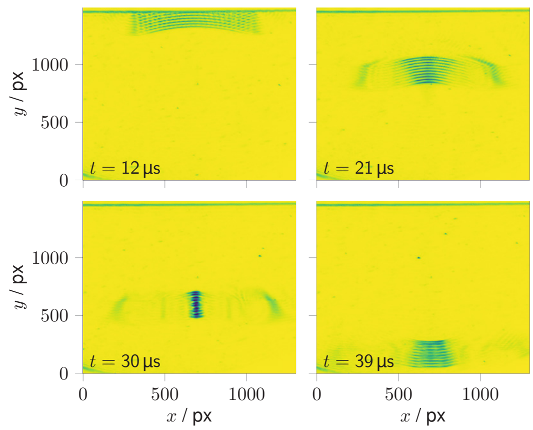

Fig. 2: Schlieren images showing the emission of the array focussing acoustic waves to a given point.

channel. The only circuit components that are required for each channel of the interface are thus the low-side gate driver and a single capacitor, leading to a minimalistic realisation of the array driver circuitry.

Evaluation

To evaluate the performance of the array interface, a 64-element phased array by *Imasonic* is used to excite acoustic waves in water. The resulting acoustic emission is visualised using schlieren imaging [3]. As an example application, a binary burst signal (10 Periods) with a frequency of 1 MHz is applied to the array. The delay times between the elements are chosen to focus the waves to a point with a distance of 30 mm in front of the array's surface. Figure 2 shows that the developed interface is able to generate the signals according to the specification and realise a successful focussing. Other beam-forming methods, such as a directed emission of planar waves with a given angle, are also evaluated successfully.

Conclusions

An interface to generate arbitrary, binary signals for a 64-element phased ultrasonic array is realised using four clock-synchronised microcontrollers and programmable input/output logic. No specialised analogue components or reconfigurable logic devices (FPGAs) are required. The presented approach can be extended to arrays with a higher number of elements by increasing the number of controllers or using an updated version (RP2350B, *Raspberry Pi Foundation*), which provides 48 programmable outputs. Future work will include an extension for simultaneous signal acquisition to realise full ultrasonic imaging capabilities.

References

[1] S. J. Rupitsch, *Piezoelectric sensors and actuators: fundamentals and applications*. Berlin: Springer, 2019, ISBN: 9783662586013.

[2] *RP2040 datasheet*, Raspberry Pi Ltd, 2021.

[3] T. Hetkämper, K. Koch, *et al.*, “Phase-preserving methods to visualise ultrasonic fields with schlieren imaging,” *tm - Technisches Messen*, vol. 90, no. 2, pp. 103–112, 2023.