IRP Phase 2: Application Specific Processor

Rufaro Manjala

Student ID: 843754953

COMPSYS 701: Advanced Digital Systems Design

Executive Summary

This was a research project to explore the concept and capability of application specific processors (ASP). Providing the benefit of improved performance and parallelism, an ASP for peak detection was made as part of this project. The ASP was made configurable, able to take instructions or data. It could detect both negative or positive peaks with an additional mode to turn off detection. This particular ASP was also made such that it could be implemented on a TDMA-MIN. In the end it proved to have supportive results that came out as expected, proving it's feasibility.

Introduction

Within digital systems there can be multiple components to enable successful operation. Within the system there could be certain repeated tasks done to data as it flows through. Instead of having a central processor needing to set aside other tasks and call a procedure or subroutine repeatedly, we can make use of what is called an application specific process (ASP). ASPs provide the benefit of better performance for specialized tasks, parallelism, and potentially lower energy consumption [1]

This document entails the research and development of an ASP that was finally used as a digital to analogue converter with peak detection capabilities

Requirements

This ASP has been developed with requirements to meet as well as adhere to constraints. These include the following:

1. The ASP must be able to integrate into a TDMA-MIN with a network interface that has 8 bits for addressing.
2. The ASP will be able to receive data and instructions that are 32 bits long
3. The ASP is configurable by another external processor which can send instructions along the paths of the TDMA
    a. Configurations include peak detection modes. There will be 3 for negative peak detection, positive peak detection, and no detection.
4. Synchronous, operations happen predictably in time with system clock
5. The data inputted into ASP is then able to be sent to an audio codec that can be outputted
6. The ASP will pass the peak value identified to be displayed

Principles of operation

Within this section, an overview of how the ASP works is provided, including an explanation on resources used as well as detailing of the data path.

Hardware and Software

Digital systems make use of a combination of hardware and software resources. Hardware that allows the implementation of physical connection, processing, conversion and reconversion is required.

Terasic's DE-1 SoC FPGA board is used for this. It proves to be particularly suited for our application as it has audio components such as audio input, audio output and audio codec. The programming instructions as well as the configurations of the connections were loaded onto this board once they were made.

Software within this project in defined the behaviour, digital connections as well as the data structures used. VHDL was used to describe and create the digital systems.

Datapath Overview

Below is a graphic showing an overall look of the ASP components and data path. The ASP for the most part is all enclosed in one component that has a 40-bit input. These 40 bits includes an 8-bit address and 32 bits that could be audio data that has been captured and processed by prior stages before being ready to be outputted, or it could be an instruction sent by another processor to configure the operation. Within the ASP the data will be processed, checking different parts of the 32-bit data, storing values if they meet conditions set within the program code. The data will then be outputted, going to an audio codec to be reconverted back to analogous data that can viewed or heard as sound waves. It must be noted that this ASP does not alter the data passed into it, but merely stores values it detects (e.g peak value) and then passes the data along. The stored values will be displayed using a seven-segment display that will be connected to this ASP.

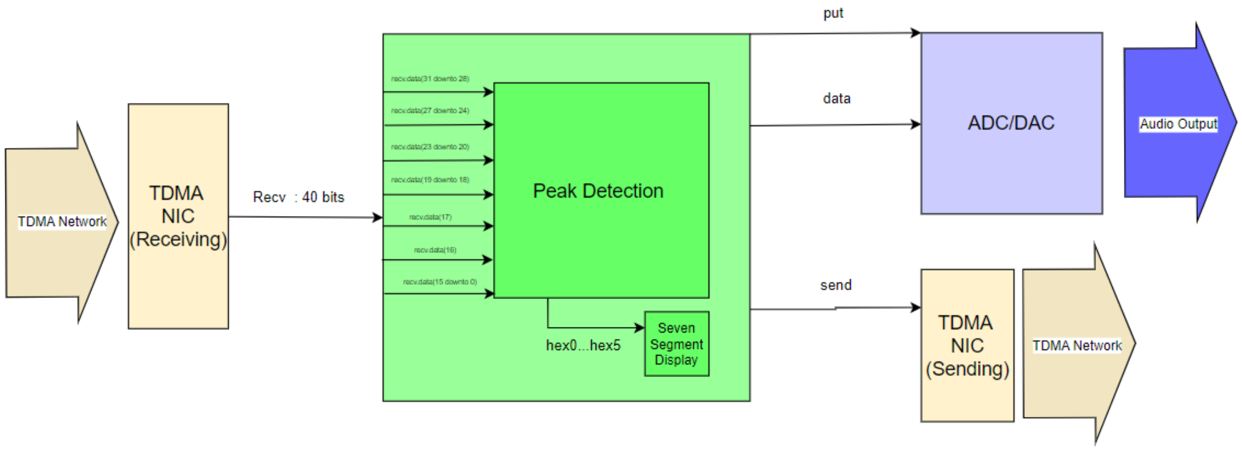
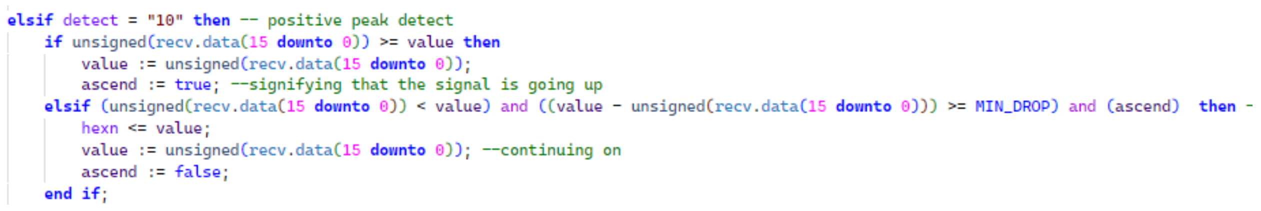


Figure 1- ASP Datapath

Peak Detection

Peak detection is the identifying of a point in a provided set of continuous data where there is a change in the gradient polarity (positive-to-negative or negative-to-positive). This processor operated with sound as data, so it was identifying peaks in amplitude from a sound wave that was inputted into the digital system.

In order to identify the peak, the ASP will be constantly comparing the last value processed with the new one being inputted. Depending on the configuration it will be watching for some criteria. For example, if the processor is configured for positive peak detection; the processor will wait until it recognizes the signal values represent a positive slope. From there it will wait until it reaches the value that is greater than all previous and is greater than the next. This will prove to be the peak of the signal. From here it will be displayed. Referring to the VHDL code will further help understanding, a snippet of the detecting algorithm has been included in this report.



```
elsif detect = "10" then -- positive peak detect
    if unsigned(recv.data(15 downto 0)) >= value then
        value := unsigned(recv.data(15 downto 0));
        ascend := true; --signifying that the signal is going up
    elsif (unsigned(recv.data(15 downto 0)) < value) and ((value - unsigned(recv.data(15 downto 0))) >= MIN_DROP) and (ascend)  then -
        hexn <= value;
        value := unsigned(recv.data(15 downto 0)); --continuing on
        ascend := false;
    end if;
```

Figure 2-snippet of detection algorithm

Further Discussion on Functionality and Instructions

Ports

The ASP has multiple ports for receiving and sending data to and from itself It has two inputs and nine output ports, the following is an explanation of each one

Clock (input): The ASP is synchronous, this means it functions predictably in time with other components of the overall system. This input receives a standard logic type value that is changing constantly over time. When this input toggles and goes from zero to one (or false to true) this represents a rising clock edge. This edge is used to begin one tick

Recv (input): This port represents the TDMA receiver port for this ASP. Data passed from another node will arrive here as an array holding two values, the address and the data. Parts of the received data will be used for processing within this ASP. See ___ for information of data format and instructions

Send (output): Data to be sent along the TDMA will be added to this port.

Put (output): Output port controlled at the top-level design of system. This port is set high when data is to be allowed to pass into the audio codec of the system to be converted back into an analogous signal.

Data (output): received data will go out from here and be passed to the audio codec to become an analogous signal. It must be noted that this ASP does not do much to alter or change the data coming in compared to what goes out. The only difference is that the data output no longer holds the most significant part signifying the type of data it is. See the ___ for more information of data format and instructions

Hex0 → Hex5 (output): All these outputs behave the same behaviour. These ports will send a 7-bit value that a seven segment display module will use to display a character. This ASP's detected peaks will be displayed on a seven-segment display.

Signals

Hexn: a 24-bit unsigned signal, this value is what will be fed into a connected seven-segment component which will then go on to discern how to display the value. Every 4 bits of this value represents one character to be displayed by the seven-segment display. Thus the connected seven-segment component is able to have up to 6 characters or digits displayed from the data this signal inputs into it.

Enable_0: a single bit that when high signifies that it is data for/from the left channel being processed

Enable_1: a single bit that when high signifies that it is data for/from the right channel being processed

Detect: a 3-bit signal that signifies what detection mode will be used by the ASP. There are 3 detection modes

- No detection – The ASP does not detect any peaks and simply lets the data pass through
- Positive peak detection – the ASP detects when there are positive peaks in the signal, and has the value of these peaks displayed on the seven-segment output
- Negative peak detection - the ASP detects when there are negative peaks in the signal, and has the value of these peaks displayed on the seven-segment output

This signal is configured based on the instruction input into the ASP. See__ for further information on data format.

Processes

There are 3 processes used within the ASP for configuring, detecting and outputting. All the processes are triggered by a rising clock edge from the clock input. Referring to the VHDL will provide for more understanding on the contents of these processes.

Seven Segment Component

Included in this project, is a seven-segment display component. This component displays any values passed to it from the ASP. The values displayed will represent a negative or positive peak. The component is a multiplexer that will output values based on the input value from the ASP (more specially the “hexn” signal). Each bit outputted from this component represents a physical segment of the display that will be turned on or off to make a character or digit. Below is a figure showing an example of how an output from this component will be seen. It is important to note that setting a bit 0 is what will turn on the corresponding segment.

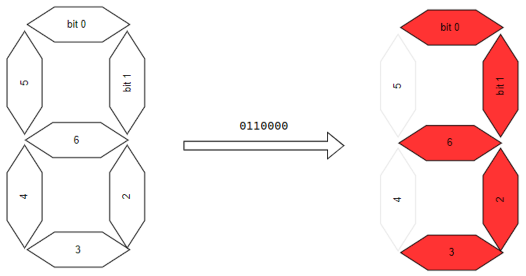


Figure 3- Seven segment example

Instruction set

Below is a graphic showing the data format of data that is inputted to the ASP, and different possible types of data that can be input

	Bits																		
	31	*30*	*29*	*28*	*27*	*26*	*25*	*24*	*23*	*22*	*21*	*20*	*19*	*18*	*17*	*16*	*15*	*..*	*0*
Data_Audio	1	0	0	0	Destination				Reserved							Channel	16 bit signed integer		
Configure_Instr	1	0	1	1	Destination				Next				Detect		Enable	Channel	Unused		

Table 1- Data input formats

The four most significant bits denote the type of data being. The *Destination* bits are optional to be used for implementing the network interface for the TDMA. The *Next* bits hold the address of where the data is to go next after being processed at the current node. The *Detect* bits are used to configure the ASP to one of the available peak detection modes. *Channel* bit denotes whether this data is for the left or right channel. A zero for the *Channel* bit signifies the left, while a one is the right channel. For the data audio format, the digital value given to the audio is in the least significant 15 bits.

The following table is to express the different possible values for the “detection type” part of incoming instructions.

Detect value	**Mode**	**Explanation**
“00”	No detection	No peak will be detected by the ASP
“01”	Positive peak detection	The positive peak value will be detected
“10”	Negative peak detection	The negative peak value will be detected

Table 2- Detection modes and bit values

Performance and Results

<<Unable to include in time of submission>>

Summary

Application Specific Processors prove to be an effective concept to implement to allow for processing data away from a central processor. With the use of an Intel SoC, FPGA, VHDL programming language and digital design concepts, it has been shown one can create a task. In this specific case an ASP that can successfully detect the peak of a signal whilst fitting the requirements of being within a TDMA-MIN network of other nodes that can include other ASPs.

References

[1] “techdesignforums,” [Online]. Available: https://www.techdesignforums.com/practice/technique/accelerating-the-implementation-of-application-specific-processors/. [Accessed May 2023].