Team 10

Matthew Arceri

Vasundhara Rawat

Will Sutton

Philip Wolfe

FPGA Clapping Pattern Detector

Final Presentation

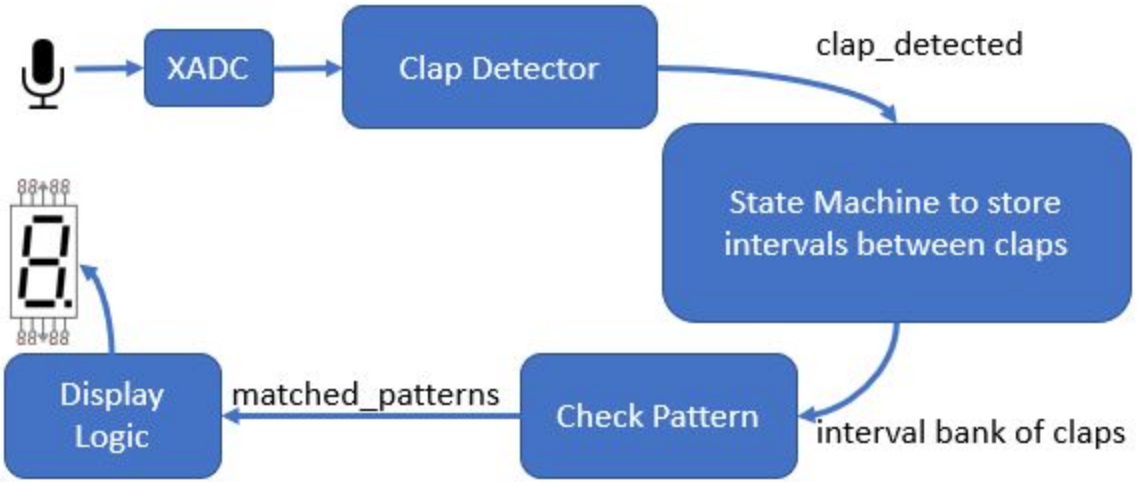
Goal:

Detect and reject certain clapping patterns and provide visual feedback.

Outline

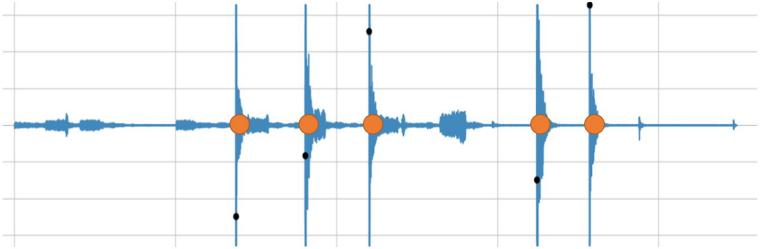
- Design Architecture
- Code Components
  - clap_detector
  - clap_FSM
  - min_not_zero & div_by_min
  - pattern_checker
  - 7-seg display and LED clap buffer
- Area, Timing, and Power
- Schedule progress
  - progress
  - next steps

Architecture - Top Level

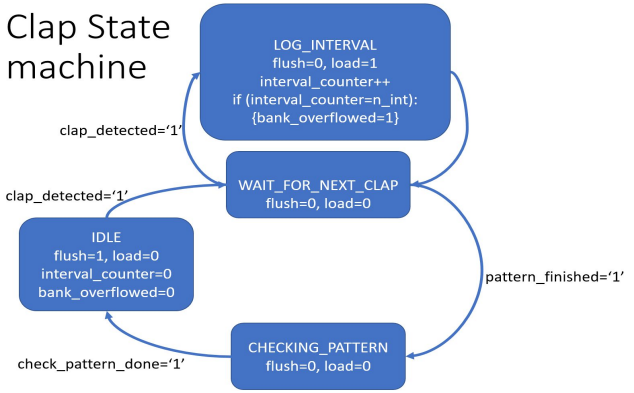


Architecture - Submodules

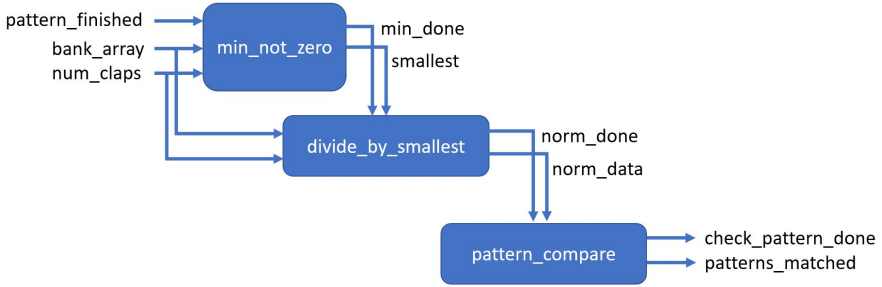
Clap Detector

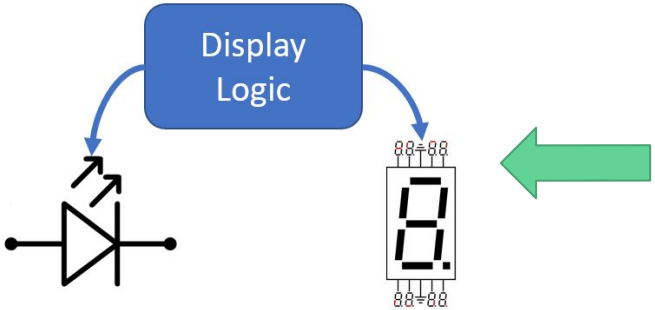


Clap State machine



check_pattern overall arch



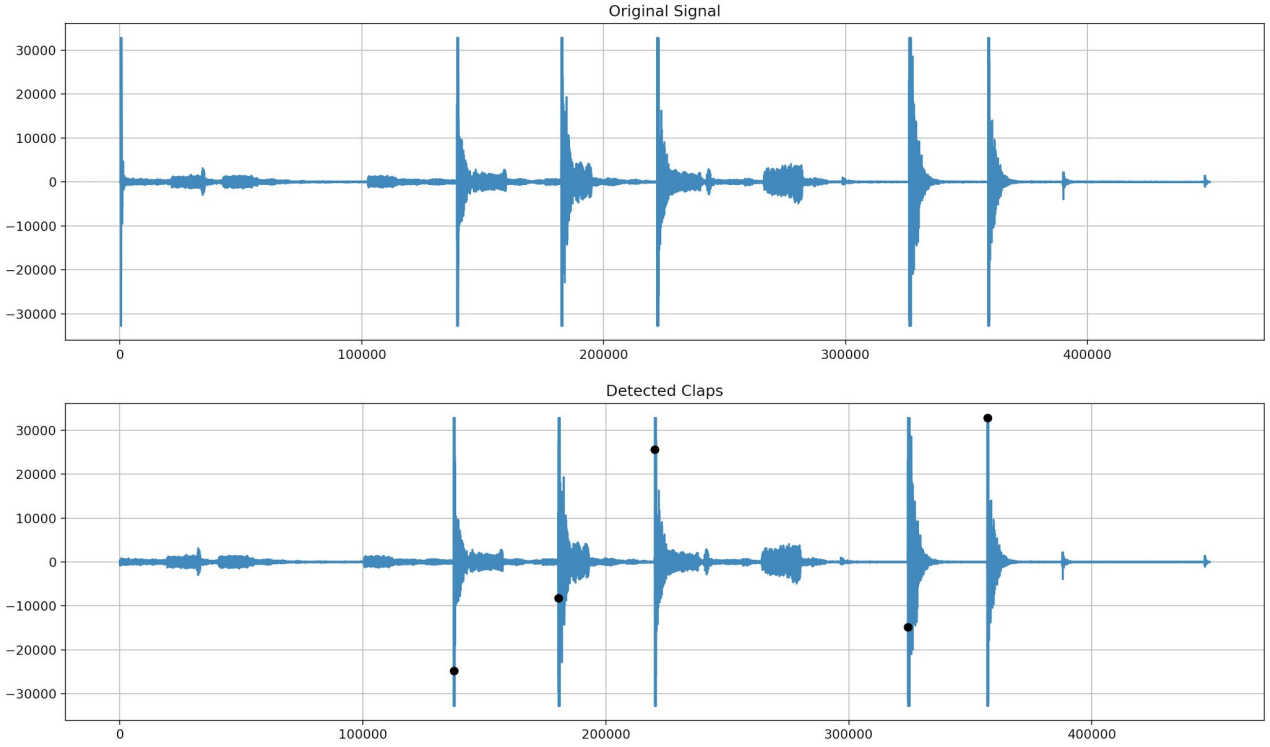


Code Components - Clap Detector

High level: python (Robust to noise)

Low level: VHDL (Preliminary algorithm, work in progress)

Code Components - Results for robust clap detection



Code Components - Clap Detector Python Implementation

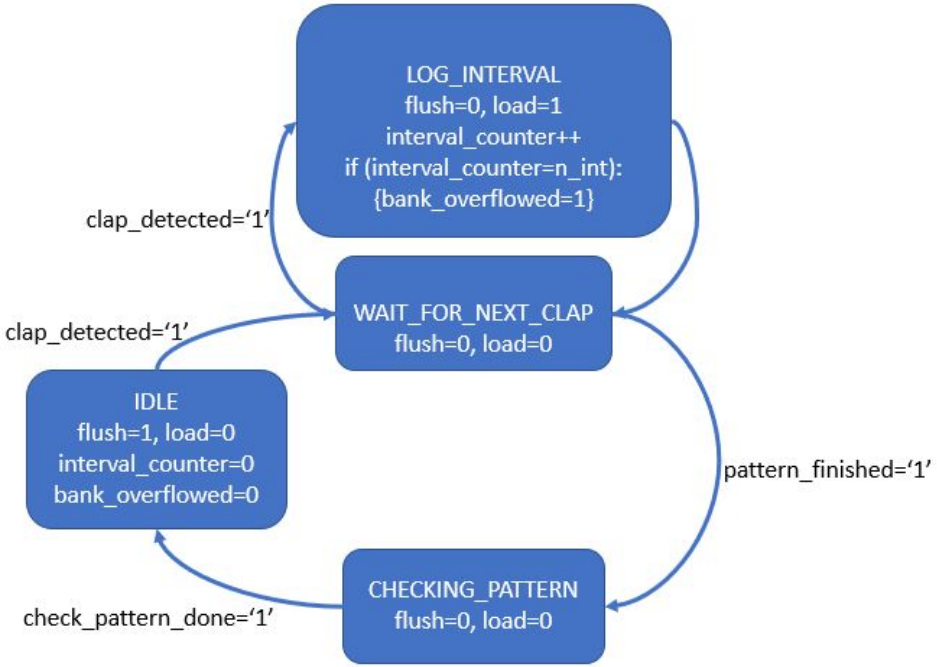
- Sliding window
- Compute short term average (mean of previous 20 samples)
- Compute long term average (mean of previous 500 samples)
- Check if short term average is at least *k* higher than long term average
- Check for clap length (at least *m*)
- Combine continuous detections

Code Components - Clap Detector VHDL Implementation

- Sliding window
- Compute short term sum
- Check if sum is greater than a given threshold *k*
    - Equivalent to comparing to average where threshold is k/n

Code Components - Clap FSM

- Manage storing time intervals between claps
- Shift register storage/data structure
- State machine logic
- Signals when pattern has finished
- Manages clearing buffer between patterns

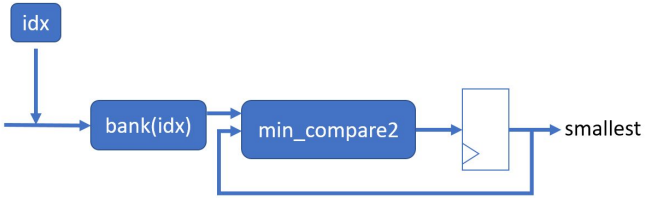


Code Components - Min_not_zero & Div_by_min

Goal: normalize data w/ respect to time

Method: divide interval bank by smallest (nonzero) interval

- Min_not_zero: find smallest nonzero interval
  - Sequentially iterate through interval bank comparing using 2-input compare module



- Div_by_min: divides each interval by the smallest interval given by min_not_zero
  - Sequentially iterate through interval bank dividing each by the smallest interval

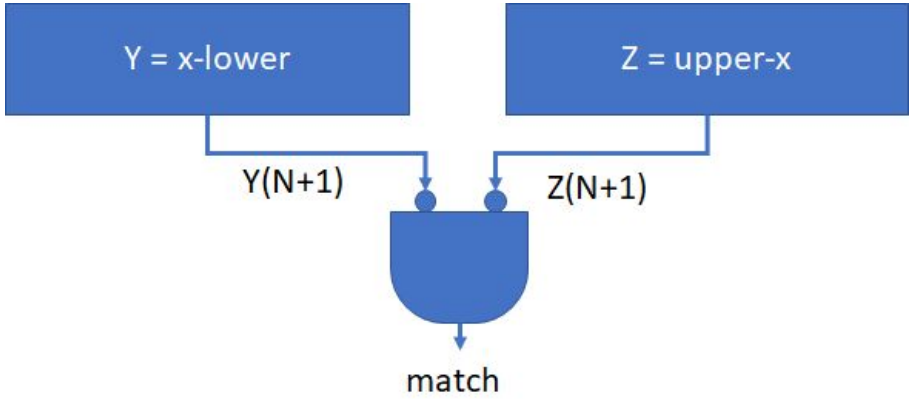
Code Components - Pattern Checker

State machine controls taking each pattern & feeding it into one boundary comp instance

- Saves significant area
- Currently does one, can possibly do two compares per clock cycle

Boundary Comp:

- Compares every bound in parallel
- For small length patterns this doesn't take much more area then sequentially checking



Code Components - Display Logic

- 7-segment manager
  - Combinational logic to converts detected pattern number & system state into visual feedback
  - All states displayed on single display digit
  - State decoding to determine output:
    - IDLE = last matched pattern (1,2,3,4)
    - WAIT_FOR_NEXT_CLAP = “L”
    - “n” for LOG_INTERVAL and CHECKING_PATTERN
  - Button debouncing was required for testing
- LED clap buffer
  - LED pattern to display presence of a clap that resets when a matched pattern is detected.
  - Shifts LEDs down every time a clap occurs
  - Clears buffer when pattern_finished from state machine = ‘1’

Area Utilization, Timing, & Power Comparison

TLD	CLB LUT Utilization	CLB Registers	WNS	Power
min_not_zero.vhd	52	33	7.894ns	.513W
pattern_compare.vhd	109	73	7.736ns	.521W
clap_FSM.vhd	Clock Dep.	Clock Dep.	Clock Dep.	Clock Dep.
divide_by_min.vhd	In progress	In progress	In progress	In progress
clap_detector.vhd	In progress	In progress	In progress	In progress
Top_level.vhd	Dependent on submodules	Dependent on submodules	Dependent on submodules	Dependent on submodules

Schedule Progress

Module	Progress	Assigned
Clap_detection	Finished, needs to be tested	Vasundhara/Matt
clap_FSM	Finished and tested, synthesis dependent on final clock freq	Philip
min_not_zero	Finished, tested, synthesized, implemented	Philip/Vasundhara
div_by_min	Divider finished & tested, need to test normalizing interval bank	Philip
pattern_checker	Finished, tested, synthesized, implemented	Matt
7-seg display	Finished, tested, synthesized, implemented	Will
LED clap buffer	Finished, tested, synthesized, implemented	Will

Next steps:
- finish testing clap_detector
- finish testing div_by_min
- instantiate modules in final top level file & program to Basys3

Questions?