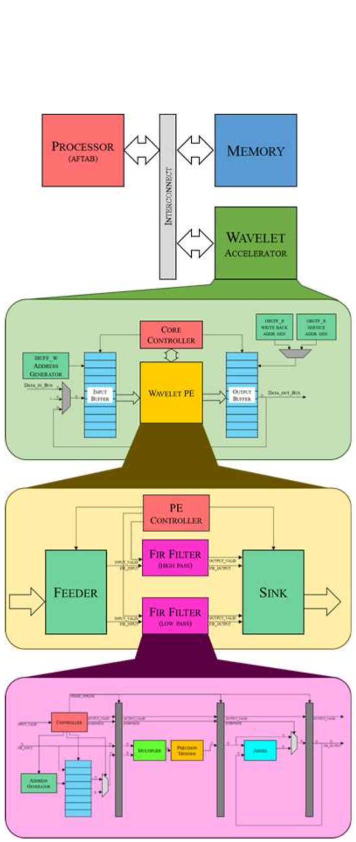
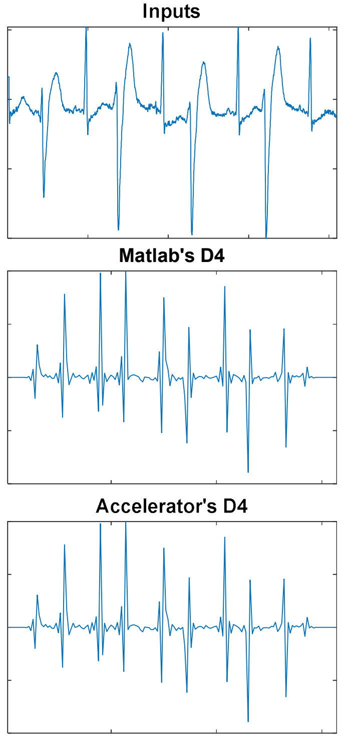


Design, Implementation, and Verification of a **Discrete Wavelet Transform** Accelerator Adjacent to AFTAB Microprocessor For Use in Bioelectronic Applications



Feb 13th, 2023

Department of Electrical Engineering at
University of Tehran

Put Together by **Moein Maleki**
As Part of Their B.Sc. Thesis Project

Advisor: **Dr. Zainalabedin Navabi**
Referee: **Dr. Seyed Kamaledin Setarehdan**

Table of Contents

Section 1

Introduction

Let your mind fly!

"Man's mind, once stretched by a new idea, never regains its original dimensions."

— Oliver Wendell Holmes

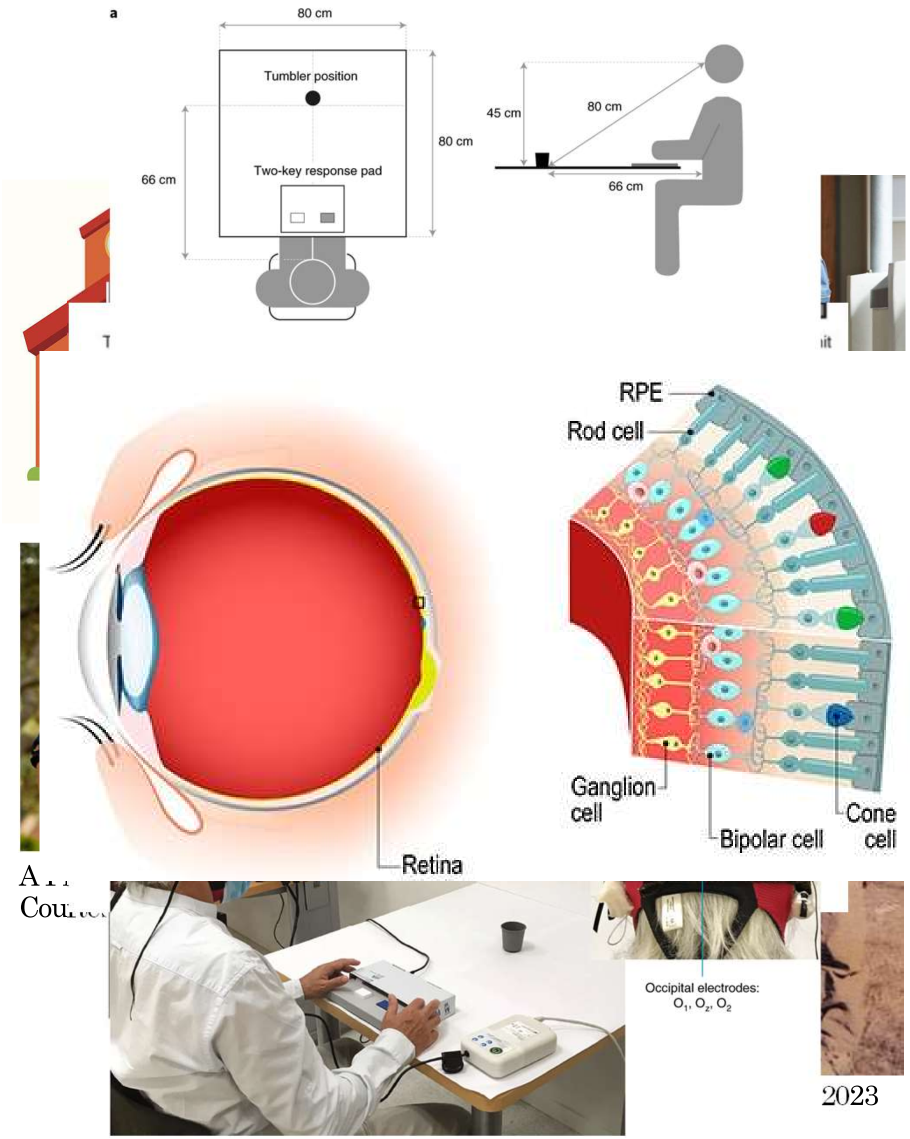
Secular Miracles

Cure for Blindness, Now Within Sight.

- Restore Vision to the Blind

- Challenges?
  - Pricey!
  - Crude Vision
  - Long Training Cycle
- Alternatives?
  - Give up!
  - Try to Find Better Solutions

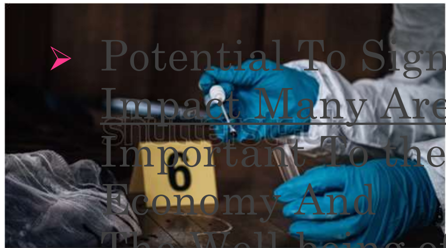
- Other Examples
  - Reverse The Effects Of Spinal Cord Injury
  - Brain Interfacing
  - Lab-On-A-Chip
  - Cardiac Pacemakers
  - Cochlear Implants
  - Glucose Monitor
  - Medical Intervention for Epilepsy, Chronic Pain, Parkinson's, Deafness, Essential Tremor



2023

Motivation. Why Bioelectronics?

- Healthcare And Medicine
- Homeland Security – Bioterrorism
- Forensics
- Protecting The Environment And The Food Supply.



- Opportunity To Dramatically Increase Synergy Between Electronics And Biology. In Monitoring, Processing, and Stimulation.
- Potential To Significantly Impact Many Areas Important To The Nation's Economy And The Well-being of its People.

Bottom line

Key Strategic Drivers
Disease Detection - Disease Prevention - Prosthetics

Semiconductor Technologies

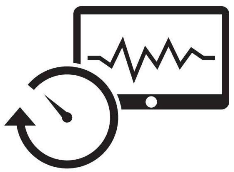
What Exactly ~~Electronics~~ Offer?

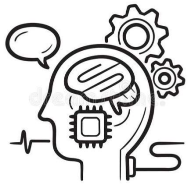
e.g., Cardiac Pacemakers, Sensory Aid Devices

e.g., Nanoscale Bioelectronic Devices
Integrated with Sensors, Actuators, and Computers

e.g., Brain-Computer interfacing.

- Real-time Measurements Decision Making.
- Power Savings wherever Possible
- Reduction in the Cost per Unit
- High-computation Power. High Throughputs.





Bottom line

There is an opportunity.
Utilize the Semiconductor Industry's Capabilities
Provide Bioelectronic-specific Tools and Systems.

A Typical Biometrics System.

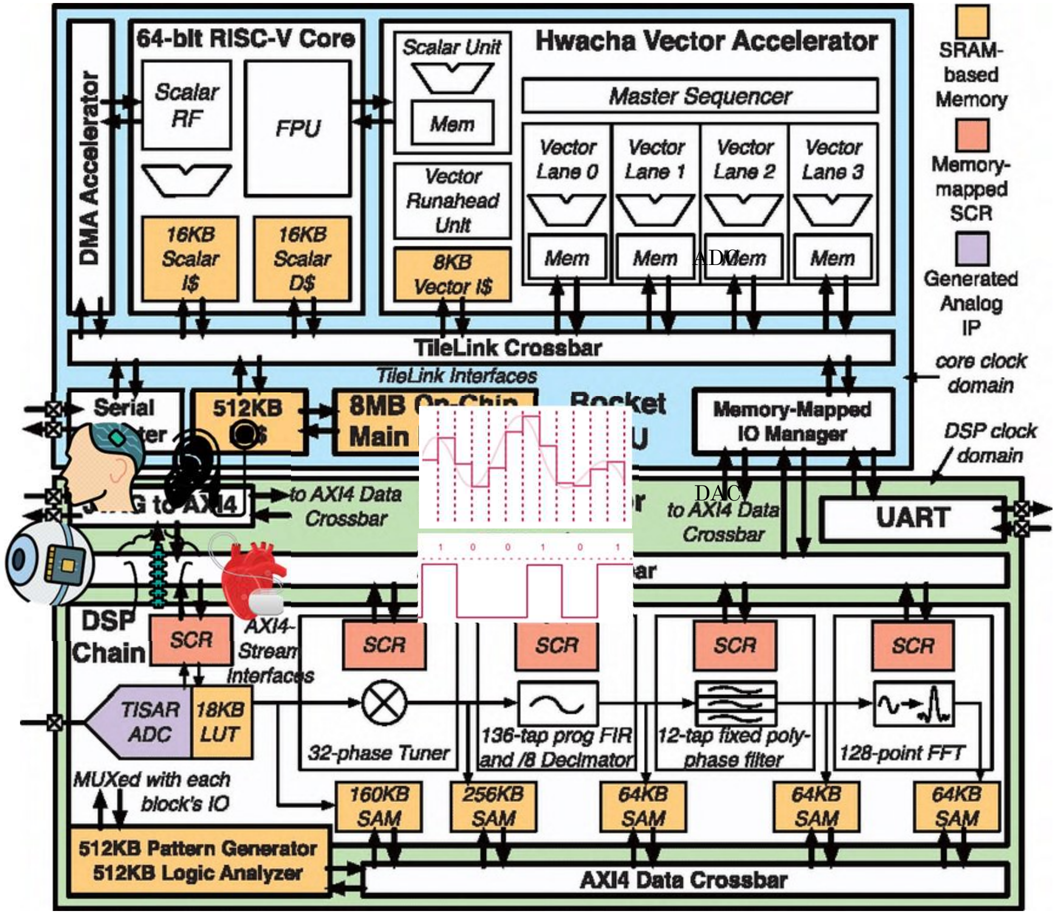
1.Signal Acquisition:
The first step involves the acquisition of signals from biological sources such as electrodes, sensors or stimulation devices.

2.Signal Pre-processing:
This stage involves removing noise, filtering, and amplifying the signals to improve the quality of the data.

3.Signal Analysis:
The processed signals are analyzed to extract relevant information, such as detecting specific patterns, calculating measurements or estimating parameters.

4.Signal Interpretation:
This stage involves the interpretation of the results obtained from the signal analysis, providing an understanding of the biological processes being monitored.

5.Signal Control and Actuation:
In some cases, the processed signals are used to control and actuate devices to influence biological processes.



How Wavelet Transform Fits.

1. WT is a Multi-resolution Analysis.

Time-frequency representation of a signal that provides information about both the location and the magnitude of its frequency components

Why is it important? →

Knowing the timing of a frequency allows for the identification of transient events and changes in the signal that are otherwise difficult to detect using other methods. This information can be used to remove noise and improve the quality of the signal for further analysis and interpretation.

2. Effectiveness in Noise Reduction and Compression

Wavelets can concentrate signal energy into a smaller number of coefficients, reducing the amount of data required for storage and analysis.

3. Sparse Representation

Wavelets often provide sparse representations of signals, meaning that most of the information is contained in a small number of coefficients.

Why is it important? →

Useful for dimensionality reduction and feature extraction.

4. Mathematical Elegance

The wavelet transform has a solid mathematical foundation, which makes it easier to understand, implement, and analyze signals.

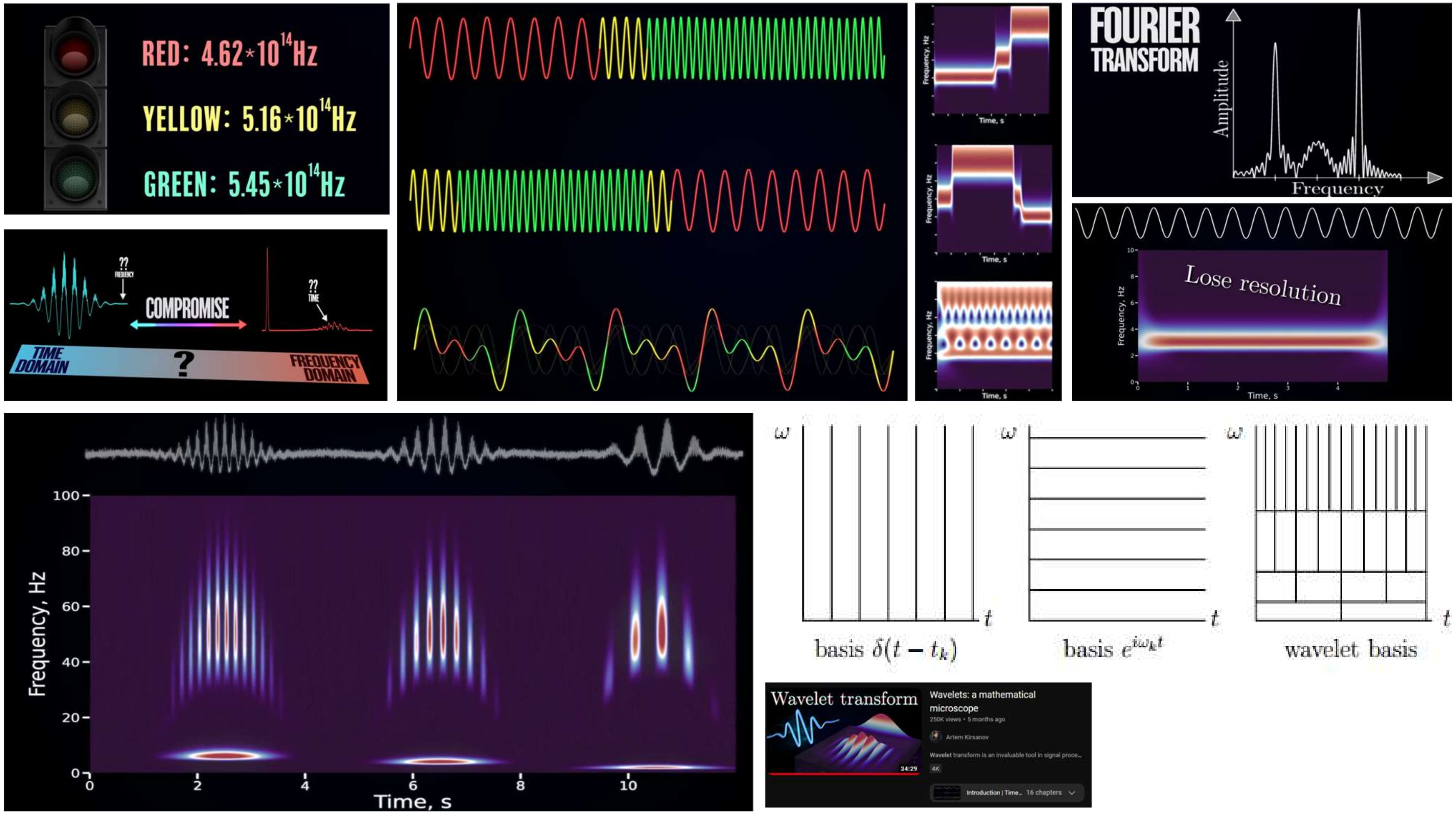
5. Handling Non-stationary Signals, and Adaptivity

Wavelet transform is better suited for handling non-stationary signals, where the frequency content of the signal changes over time.

Why is it important? →

Fourier transform assumes that the signal is stationary, which may not be the case in many real-world signals.

A Great YouTube Video: Click here!



Applications of Wavelet Transform

1. Noise Reduction/Removal
Using wavelet thresholding, WT enables us to eliminate the high-frequency noise components without affecting the low-frequency signal components.

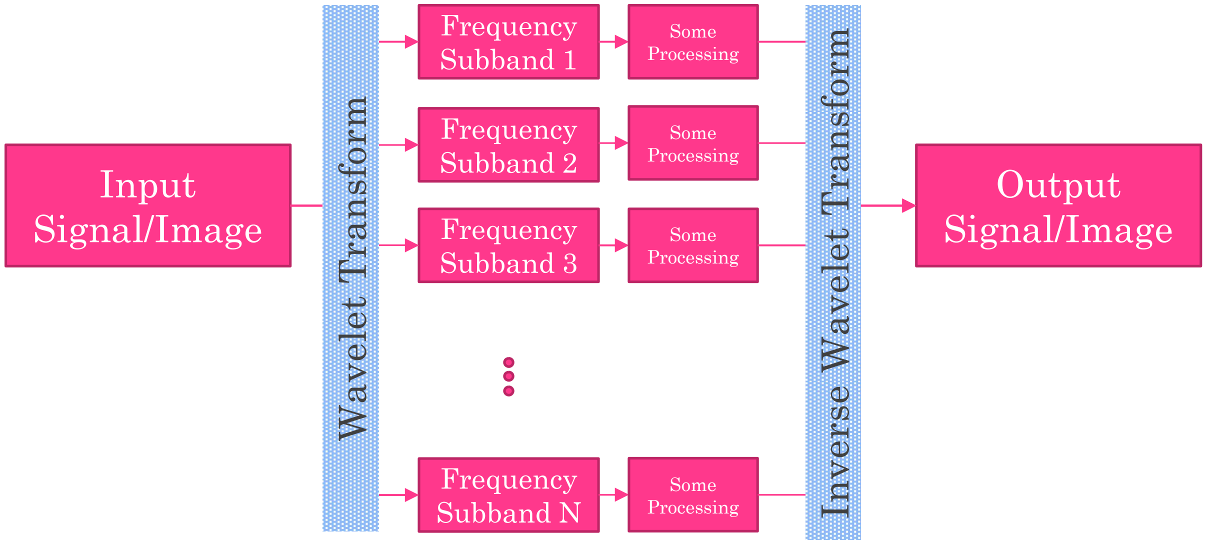
2. Image Compression
WT decomposes the signals into sub-bands of different frequencies. Each sub-band is then compressed based on its statistical properties, preserving the most important information.

3. Quality Assessment
By analyzing the decomposition coefficients, quality assessment metrics such as signal-to-noise ratio, entropy, and energy can be calculated. Can also be used to detect outliers and anomalies, and identify changes in the signal over time

4. Data Fusion
WT is used to combine information from multiple sources, improving the overall quality and reliability of the data. Frequency components can be weighted, based on their relevance or quality, before being recombined into a single signal or image.

5. Watermarking
WT is used to embed and detect digital watermarks. Helps ensure the authenticity and integrity of the biomedical signals and images.

6. Feature Extraction
WT enables the extraction of important features or patterns at different scales. This information can then be used for various applications, such as classification, diagnosis, or monitoring of medical conditions.



What is Wavelet Transform?

- Short Answer? A Mathematical Microscope.

- Long Answer?

Continuous WT Mother Wavelets:

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\tau}{s}\right)$$

Continuous WT Overarching Equations:

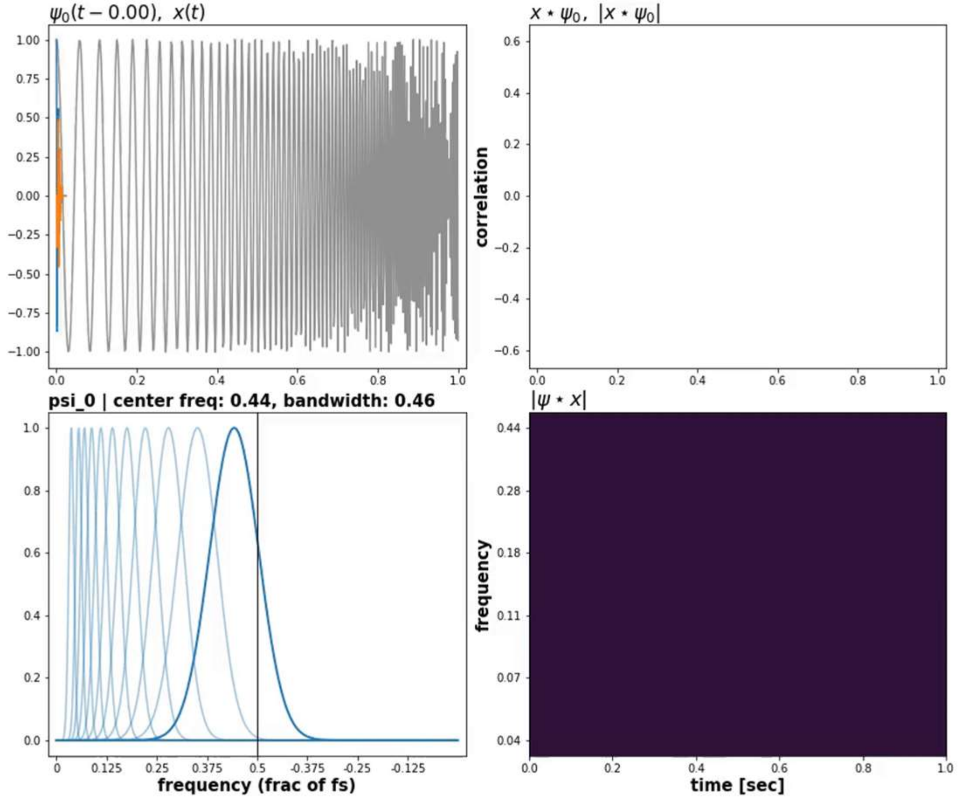
$$\gamma(s,\tau) = \int f(t)\psi^*_{s,\tau}(t)dt \qquad f(t) = \int\int \gamma(s,\tau)\psi_{s,\tau}(t)d\tau ds$$

Discrete (Fast) WT Mother Wavelets:

$$\psi_{j,k}(t) = \frac{1}{\sqrt{s_0^j}}\psi\left(\frac{t-k\tau_0 s_0^j}{s_0^j}\right). \qquad s_0 = 2, \tau_0 = 1$$

Fourier Property:

$$F\{f(at)\} = \frac{1}{|a|}F\left(\frac{\omega}{a}\right).$$



Section 2

References

for Future Improvement

Main References for Future Improvement

[1] G. Singh, S. R. Chiluveru, B. Raman, M. Tripathy, and B. K. Kaushik, “Novel Architecture for Lifting Discrete Wavelet Packet Transform With Arbitrary Tree Structure,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 29, no. 7. Institute of Electrical and Electronics Engineers (IEEE), pp. 1490–1494, Jul. 2021. doi: 10.1109/tvlsi.2021.3079989.

[2] S. A. Salehi and D. D. Dhruba, “Efficient Hardware Implementation of Discrete Wavelet Transform Based on Stochastic Computing,” 2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI). IEEE, Jul. 2020. doi: 10.1109/isvlsi49217.2020.00083.

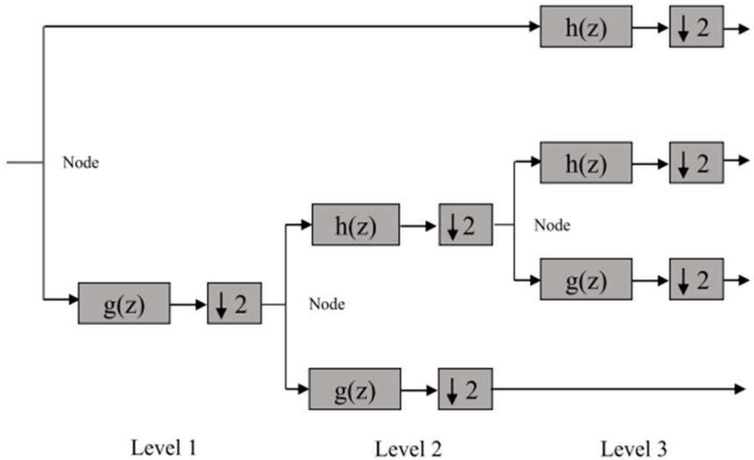
[3] M. Tausif, E. Khan, M. Hasan, and M. Reisslein, “Lifting-Based Fractional Wavelet Filter: Energy-Efficient DWT Architecture for Low-Cost Wearable Sensors,” Advances in Multimedia, vol. 2020. Hindawi Limited, pp. 1–13, Dec. 16, 2020. doi: 10.1155/2020/8823689.

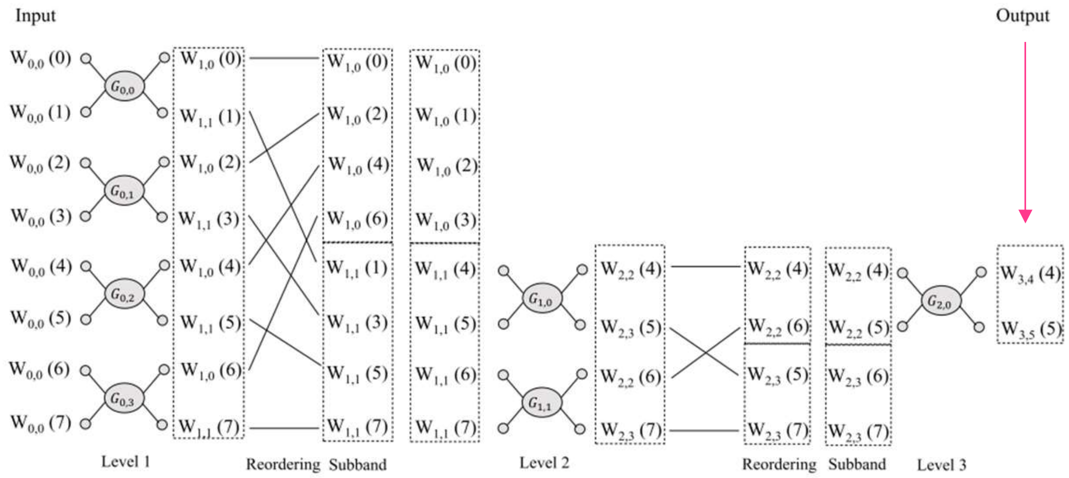
[4] R. Pinto and K. Shama, “An Efficient Architecture for Modified Lifting-Based Discrete Wavelet Transform,” Sensing and Imaging, vol. 21, no. 1. Springer Science and Business Media LLC, Oct. 26, 2020. doi: 10.1007/s11220-020-00317-z.

[5] M. Tausif, E. Khan, M. Hasan, and M. Reisslein, “SMFrWF: Segmented Modified Fractional Wavelet Filter: Fast Low-Memory Discrete Wavelet Transform (DWT),” IEEE Access, vol. 7. Institute of Electrical and Electronics Engineers (IEEE), pp. 84448–84467, 2019. doi: 10.1109/access.2019.2924490.

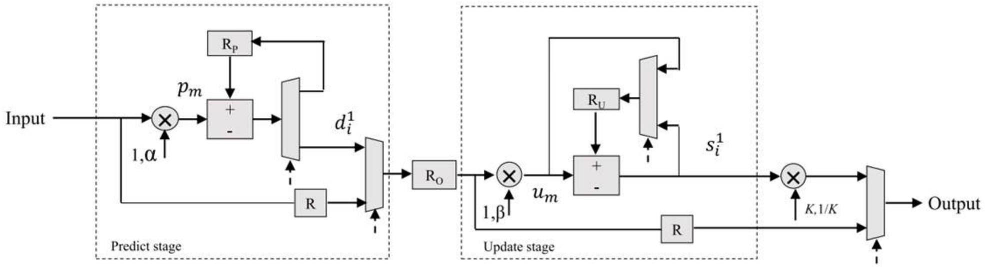
Novel Architecture for Lifting Discrete Wavelet Packet Transform With Arbitrary Tree Structure

G. Singh et al. [2021]





SFG of 3-level 8-point full DWPT decomposition with arbitrary tree.



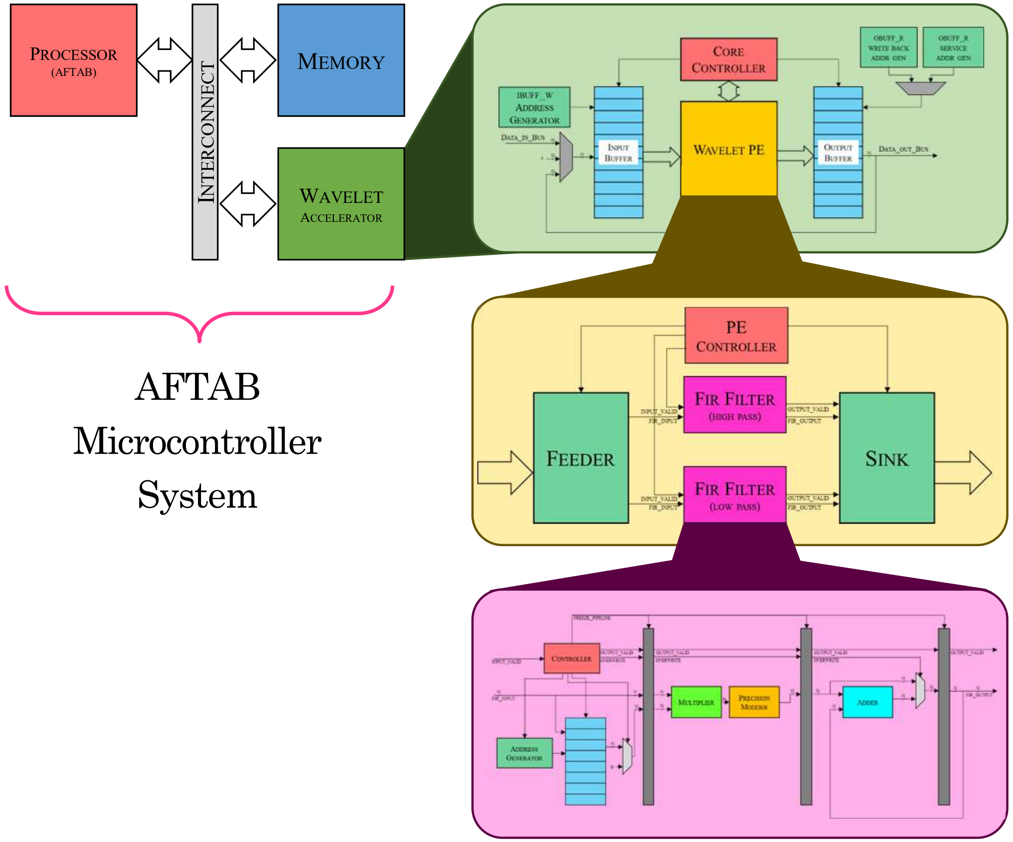
Proposed bypassed lifting-based (5, 3) wavelet filter.

Parameters	Wang *et al.* [13]	Garcia *et al.* [14]	This work
Area in μm^2 (@90nm)	16255	17327	11622
Power in Watt (@ 250 MHz)	0.0155	0.016	0.013
Gate count	2939	3133	2102

Section 3

Accelerator Design

Bird’s Eye View



AFTAB
Microcontroller
System

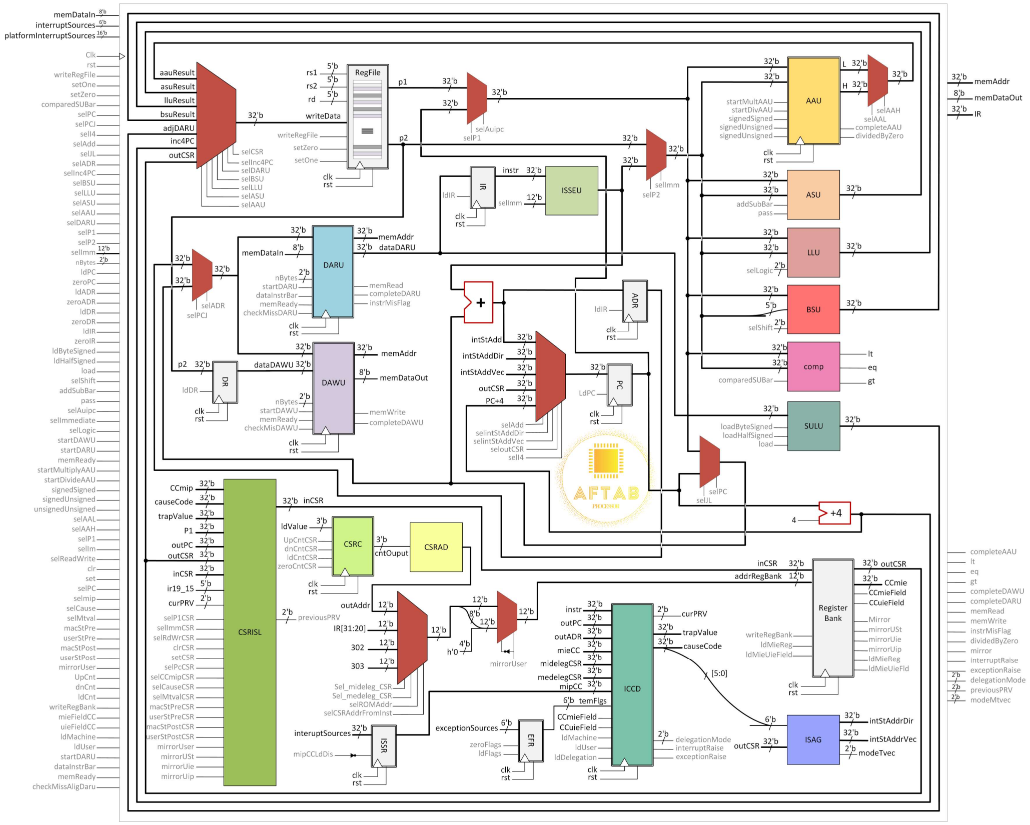
Wavelet Accelerator

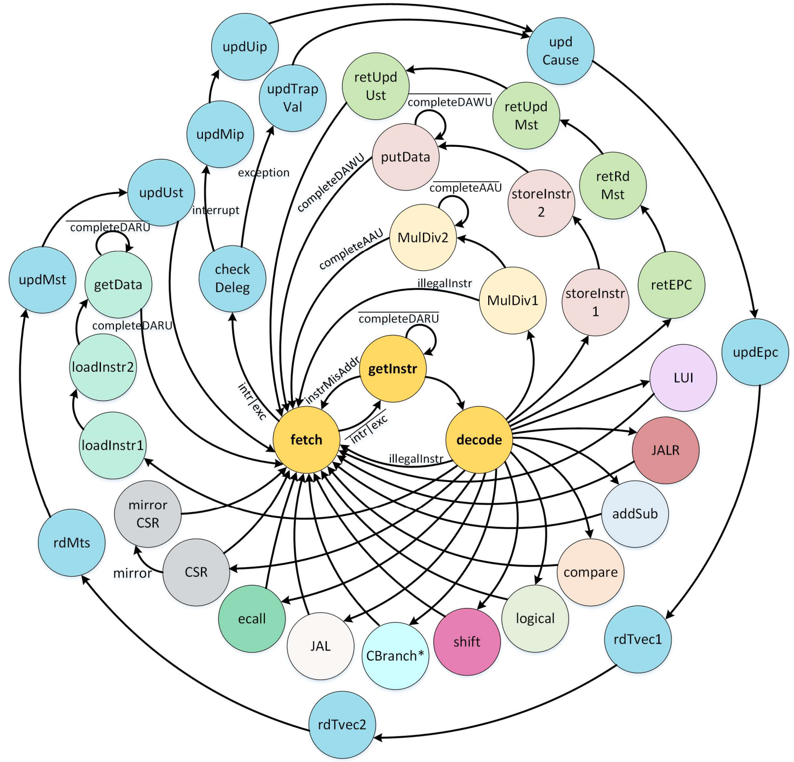
Wavelet Core

Wavelet Processing Element

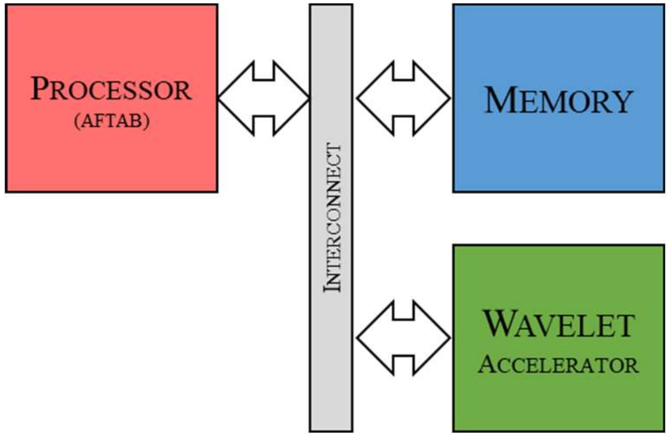
Convolution Element
FIR Filter

AFTAB Microcontroller

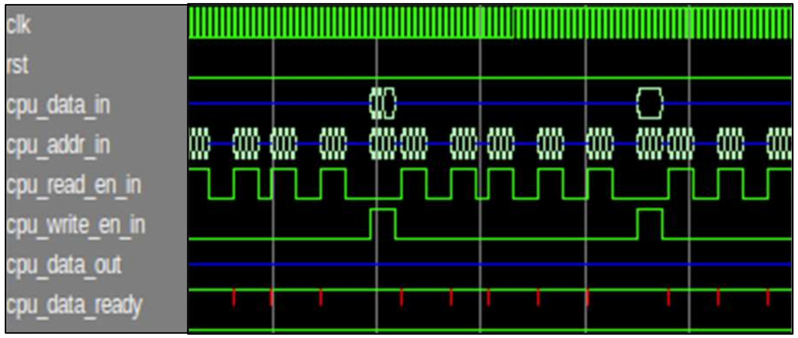




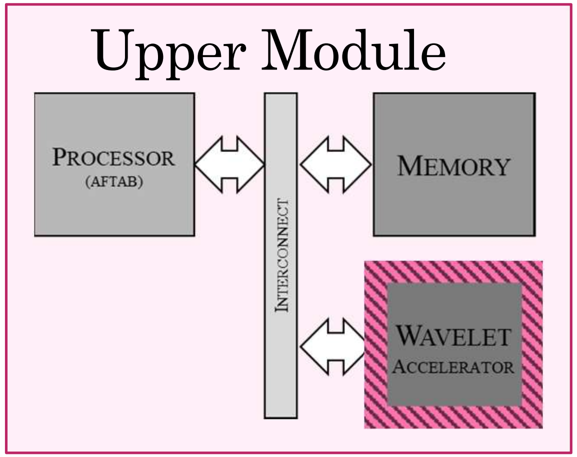
Accelerator Slave Port

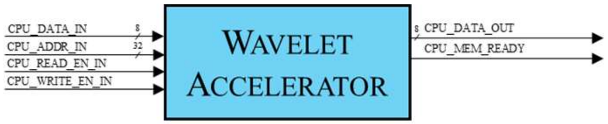


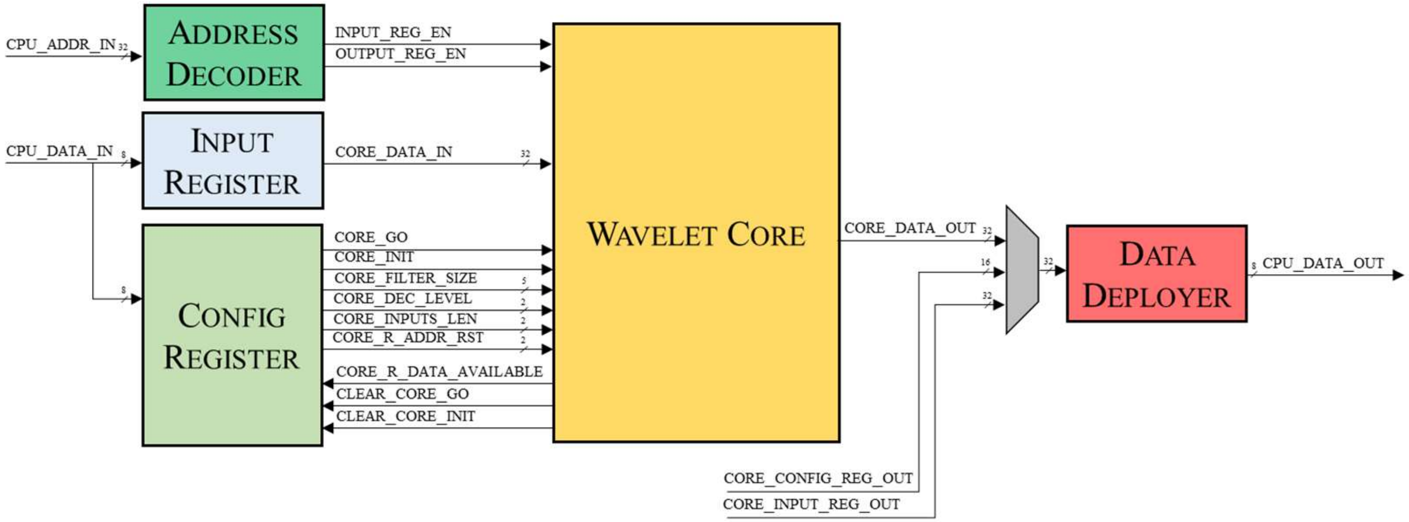
Name	Direction (I/O)	Width
CPU_DATA_IN	Input	8-bits
CPU_ADDRESS_IN	Input	32-bits
CPU_READ_EN_IN	Input	1-bit
CPU_WRITE_EN_IN	Input	1-bit
CPU_DATA_OUT	Output	8-bits
CPU_MEM_READY_OUT	Output	1-bit



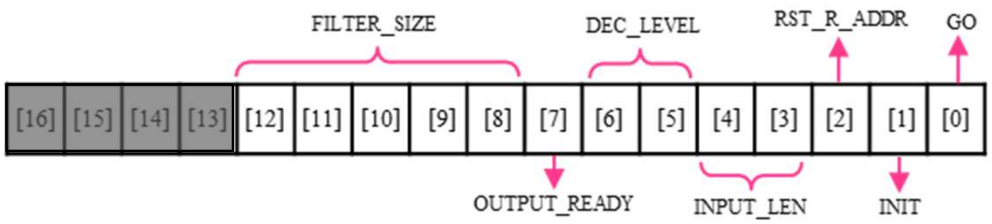
Wavelet Accelerator



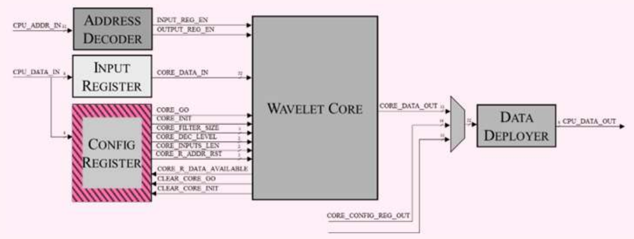




Configuration Register



Upper Module



1.GO:
Starts the DWT calculation. Signals the accelerator that the upcoming writes in the Input register are input signal data. After Setting GO, the user's writes to the config register only accept changes to the RST_R_ADDR bit. Will be Cleared when the DWT calculation is finished.

2.INIT:
Signals the accelerator that the upcoming writes in the Input register are DWT Decomposition Filter Coefficients. After Setting GO, the user's writes to the config register only accept changes to the GO bit. Will be Cleared when FILTER_SIZE times 2, number of writes to the input register are done.

3.RST_R_ADDR:
Resets the Pointer to the Output Buffer where the Next Available Output Data is located.

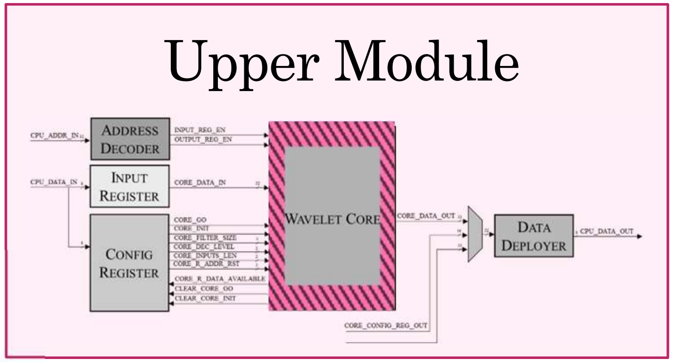
4.INPUT_LEN
Once decoded, determines the number of input signal data.
2'b00 = 256 samples
2'b01 = 512 samples
2'b10 = 1024 samples
2'b11 = 2048 samples

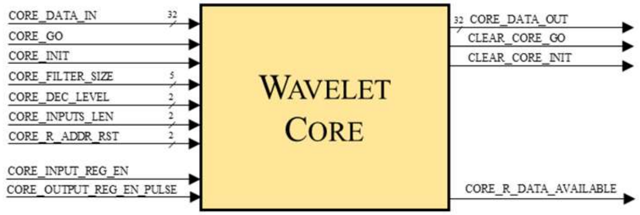
5.DEC_LEVEL
Once Incremented by One, Determines the number of Desired Decomposition Levels.

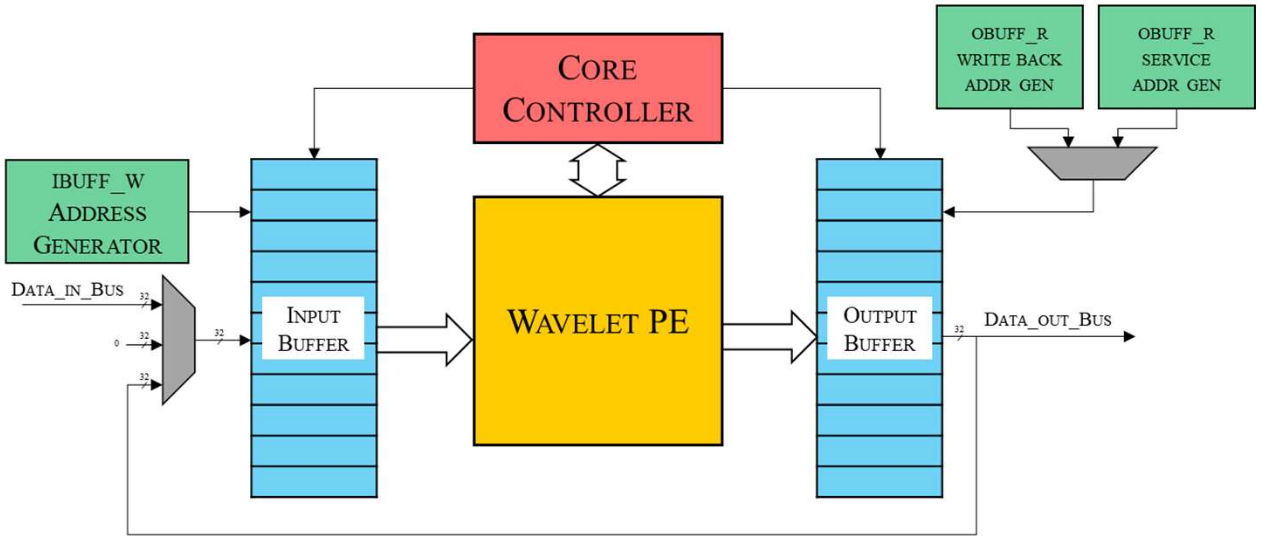
6.OUTPUT_READY
Writes to this bit from the outside are ignored. Once set by core, Conveys there exists unread data in the Output Buffer.

7.FILTER_SIZE:
Once incremented by one, Determines the size of Wavelet Decomposition Filter

Wavelet Core





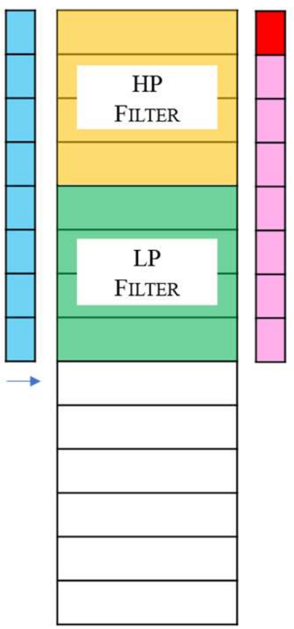


Input Buffer

Wavelet Core

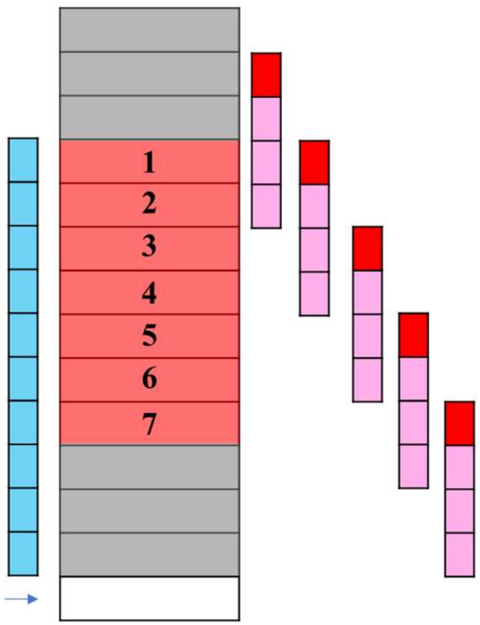
INPUT BUFFER: INIT

HOW IT'S WRITTEN: HOW IT'S READ:

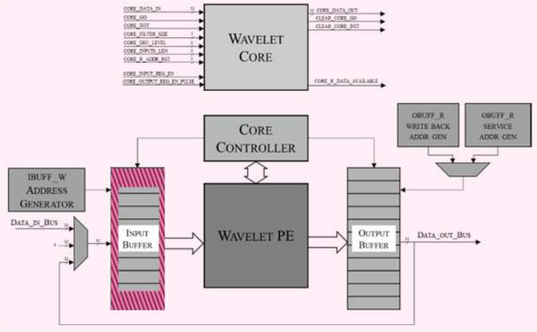


INPUT BUFFER: GO

HOW IT'S WRITTEN: HOW IT'S READ:

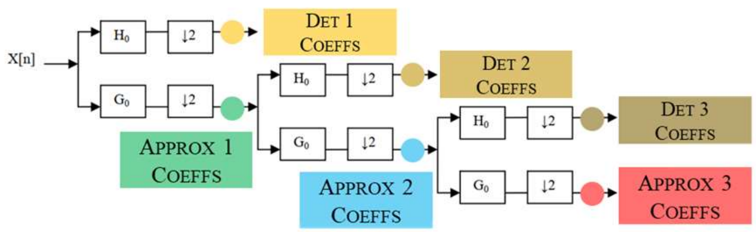


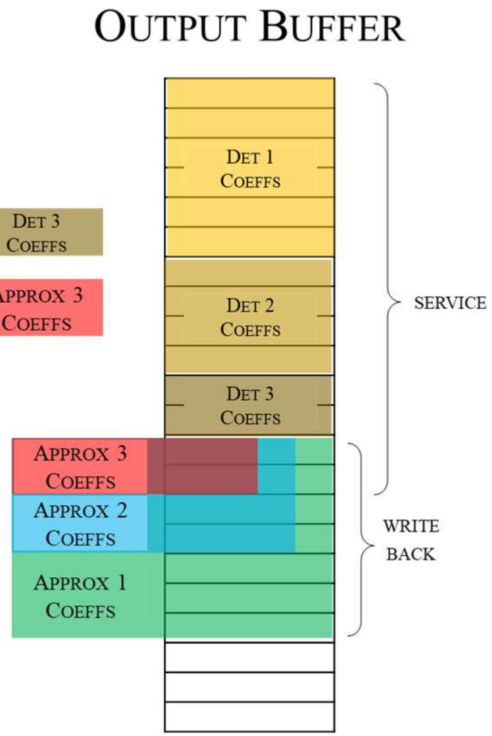
Upper Module



Output Buffer

Wavelet Core





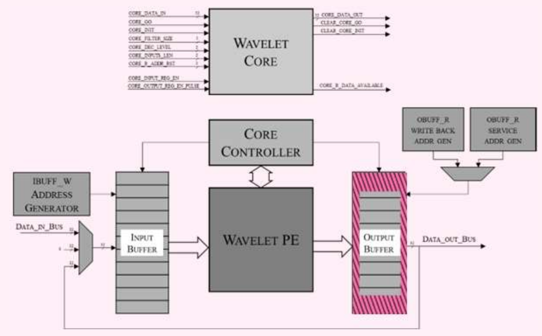
Output lengths:

Level 1: (256 + 4 - 1) >> 1 = 129

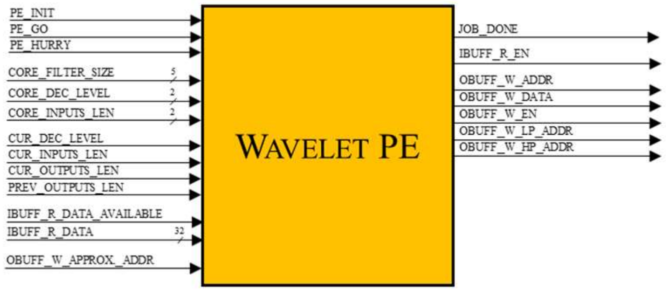
Level 2: (129 + 4 - 1) >> 1 = 66

Level 3: (66 + 4 - 1) >> 1 = 34

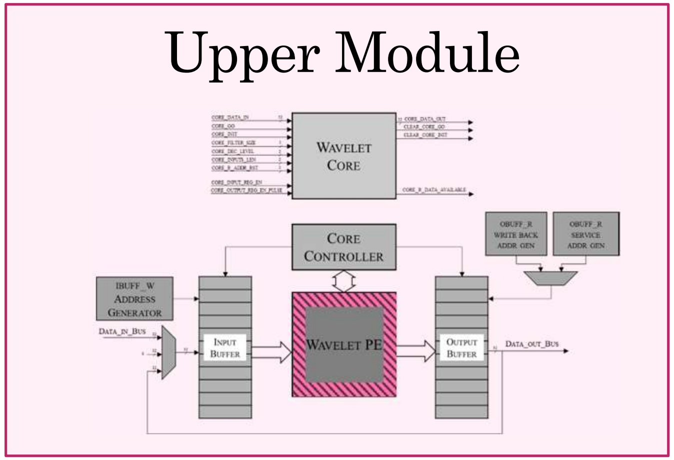
Upper Module



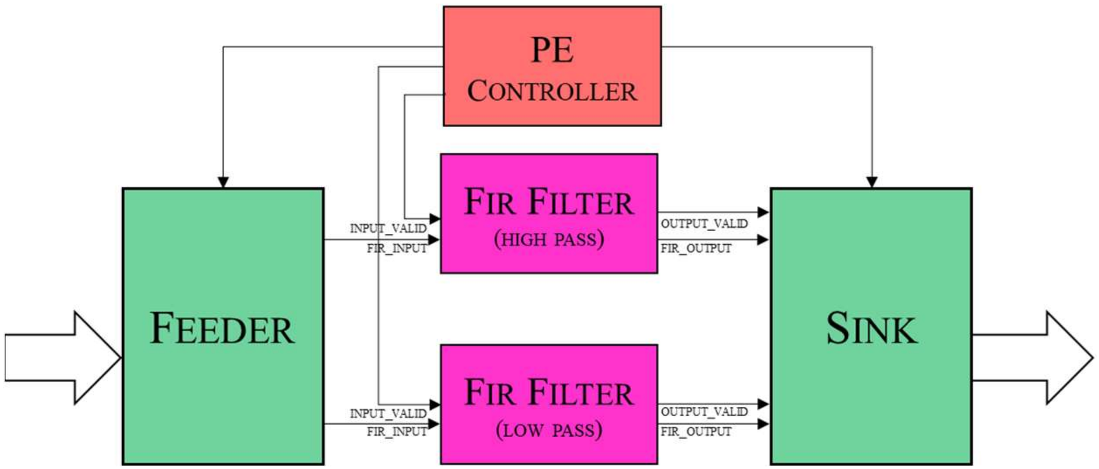
Wavelet PE



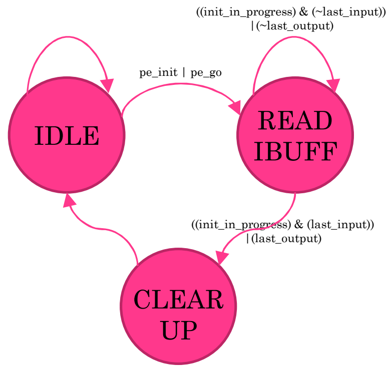
PE_INIT
PE_GO
PE_HURRY
CORE_FILTER_SIZE
CORE_DEC_LEVEL
CORE_INPUTS_LEN
CUR_DEC_LEVEL
CUR_INPUTS_LEN
CUR_OUTPUTS_LEN
PREV_OUTPUTS_LEN
IBUFF_R_DATA_AVAILABLE
IBUFF_R_DATA
OBUFF_W_APPROX_ADDR
WAVELET PE
JOB_DONE
IBUFF_R_EN
OBUFF_W_ADDR
OBUFF_W_DATA
OBUFF_W_EN
OBUFF_W_LP_ADDR
OBUFF_W_HP_ADDR



Upper Module
WAVELET CORE
CORE CONTROLLER
IBUFF_W ADDRESS GENERATOR
DATA_IN_BUS
INPUT BUFFER
WAVELET PE
OUTPUT BUFFER
DATA_OUT_BUS

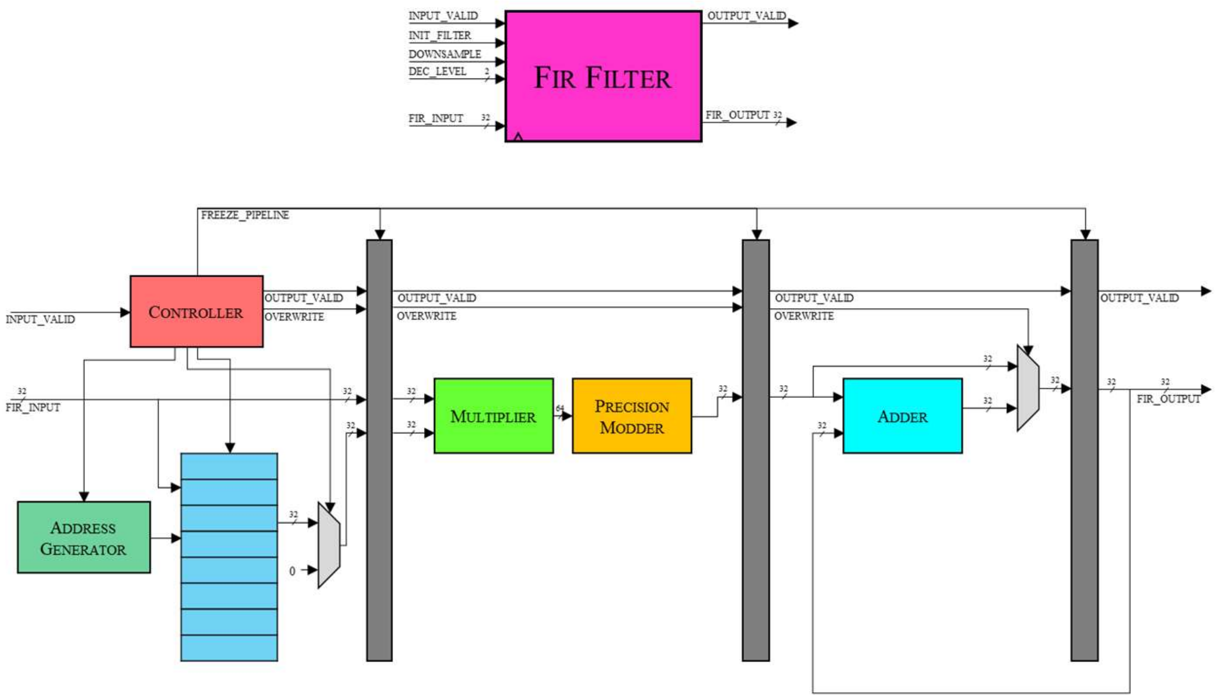


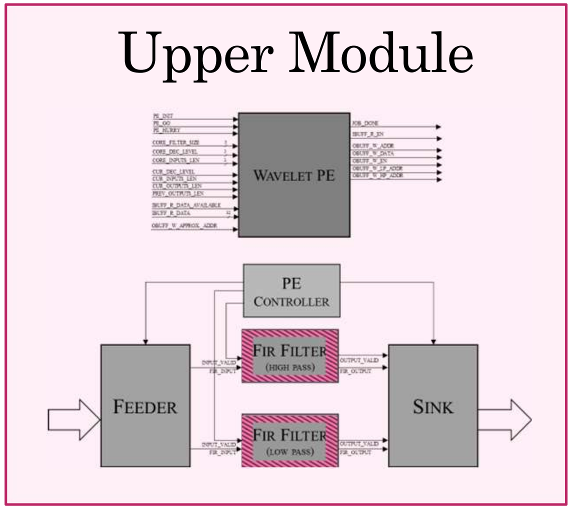
PE CONTROLLER
FEEDER
FIR FILTER (HIGH PASS)
FIR FILTER (LOW PASS)
SINK
INPUT_VALID
FIR_INPUT
OUTPUT_VALID
FIR_OUTPUT

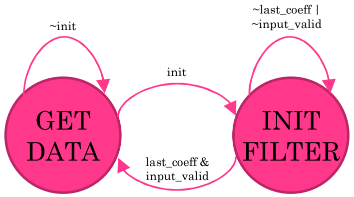


((init_in_progress) & (~last_input)) | (~last_output)
pe_init | pe_go
IDLE
READ IBUFF
((init_in_progress) & (last_input)) | (last_output)
CLEAR UP

FIR Filter


FIR FILTER
INPUT_VALID
INIT_FILTER
DOWNSAMPLE
DEC_LEVEL
FIR_INPUT
OUTPUT_VALID
FIR_OUTPUT
FREEZE_PIPELINE
CONTROLLER
OVERWRITE
ADDRESS GENERATOR
MULTIPLIER
PRECISION MODDER
ADDER


Upper Module
WAVELET PE
PE CONTROLLER
FIR FILTER (HIGH PASS)
FIR FILTER (LOW PASS)
FEEDER
SINK


~init
~last_coeff |
~input_valid
init
GET DATA
INIT FILTER
last_coeff &
input_valid

Section 4

Evaluation

Evaluation

Metrics

1. Root Mean Square Error (RMSE) Between MATLAB's Official DWT function, and the Developed Accelerators outputs

2. Speed-up Between Hardware accelerated DWT and Software Implementation of DWT

3. FPGA Resource Usage Before and After Adding the Accelerator

4. Area Comparison on ASIC Before and After Adding the Accelerator

Actions Taken:

- Developed Proprietary 1-D and 2-D DWT function in MATLAB
- Developed Proprietary MATLAB to Compare and Display the Results
- Developed Testbenches at Various Stages to Mimic AFTAB Behaviour
- Developed C Code to Perform and Run DWT Using SW on AFTAB
- Performed Synthesis Using Quartus on DE2 FPGA board
- Performed ASIC Synthesis and Compared HW Overhead

RMSE results

[MIT-BIH ECG Arrythmia Dataset]
[Max Input Amplitude: 0.1] [FS = 360 Hz]

<table>
<tr><th colspan="5">Test 1</th><th colspan="4">Test 2</th></tr>
<tr><td>Input Length</td><td>Filter Size</td><td>Dec Level</td><td colspan="2">Wavelet family</td><td>Input Length</td><td>Filter Size</td><td>Dec Level</td><td>Wavelet family</td></tr>
<tr><td>2048</td><td>32</td><td>4</td><td colspan="2">Sym16</td><td>256</td><td>4</td><td>1</td><td>Db2</td></tr>
<tr><td colspan="5">RMS Errors</td><td colspan="4">RMS Errors</td></tr>
<tr><td>D1</td><td>D2</td><td>D3</td><td>D4</td><td>Appr</td><td colspan="2">D1</td><td colspan="2">Appr</td></tr>
<tr><td>$0.0226e^{-6}$</td><td>$0.0227e^{-6}$</td><td>$0.0233e^{-6}$</td><td>$0.0238e^{-6}$</td><td>$0.1501e^{-6}$</td><td colspan="2">$0.2853e^{-8}$</td><td colspan="2">$0.2783e^{-8}$</td></tr>
<tr><td colspan="5">Test's Mean RMS Error</td><td colspan="4">Test's Mean RMS Error</td></tr>
<tr><td colspan="5">$4.85e^{-8}$</td><td colspan="4">$2.81e^{-9}$</td></tr>
<tr><td colspan="9">Mean RMS Error</td></tr>
<tr><td colspan="9">$2.56e^{-8}$</td></tr>
</table>

Speed-up results

Test 1				Test 2			
Input Length	Filter Size	Dec Level	Wavelet family	Input Length	Filter Size	Dec Level	Wavelet family
2048	32	4	Sym16	256	4	1	Db2
HW Accelerated Runtime		Software Runtime		HW Accelerated Runtime		Software Runtime	
768,760 ns		≈ 119,919,936 ns		635,620 ns		14,594,030 ns	
Test's Speed-up				Test's Speed-up			
159.9				22.9			
Mean Speed-up							
91.4							

FPGA Resource Usage and FMAX

DE2 Cyclone II EP2C35F672C6	
Combinational Function	Logic Registers
1,556 / 33,216 (5%)	456 / 33,216 (1%)
Total Logic Elements	
1,619 / 33,216 (5%)	
Total Memory Bits	
264,192 / 483,840 (55%)	
Embedded 9-bit Multiplier elements	
16 / 70 (13%)	

DE2 Cyclone II EP2C35F672C6
FMAX
61.36 MHz

Accelerator Testbench

```verilog
initial begin
    clk                 = 0;
    rst                 = 0;
    cpu_data_in         = {(8){1'bz}};
    cpu_addr_in         = {(32){1'bz}};
    cpu_read_en_in      = 0;
    cpu_write_en_in     = 0;
    filter_size         = FILTER_SIZE - 1;
    inputs_len          = INPUTS_LEN;
    dec_level           = DEC_LEVEL - 1;

    duv_system_reset();

    duv_put_configs();

    duv_put_init();

    duv_put_coeffs();

    duv_wait_init_finish();

    duv_put_go();

    duv_reset_r_addr();

    duv_put_signal();

    duv_wait_go_finish();

    duv_read_outputs();

    repeat(100) @(posedge clk);
    $stop();
end

endmodule
```

DWT Using Hardware

```c
int main()
{
    // duv_put_configs
    *((int*)0x1A100000) = 0x0300; //000_00011_0_00_00_0_0_0;

    // duv_put_init
    *((int*)0x1A100000) = 0x0302; //000_00011_0_00_00_0_1_0;

    // duv_put_coeffs
    *((int*)0x1A100004) = 0xC22E4571;
    *((int*)0x1A100004) = 0x6B12F75B;
    *((int*)0x1A100004) = 0xE34F40F4;
    *((int*)0x1A100004) = 0xEF6F823E;

    *((int*)0x1A100004) = 0xEF6F823E;
    *((int*)0x1A100004) = 0x1CB0BF0B;
    *((int*)0x1A100004) = 0x6B12F75B;
    *((int*)0x1A100004) = 0x3DD1BA8E;

    // duv_wait_init_finish
    while((*((int*)0x1A100000) & 0x0002)) {}

    // duv_put_go
    *((int*)0x1A100000) = 0x0301;

    // duv_reset_r_addr
    *((int*)0x1A100000) = 0x0306;

    // duv_put_signal
    int input_idx;
    for(input_idx = 0; input_idx < 256; input_idx = input_idx + 1) {
        *((int*)0x1A100004) = test_inputs[input_idx];
    }

    // duv_read_outputs
    int output_idx = 0;
    int output_data;
    while(1) {
        while(!(*((int*)0x1A100000) & 0x0080)) {}
        output_data = *((int*)0x1A100008);
        output_idx = output_idx + 1;
        if(output_idx == 258)
            break;
    }

    return 0;
}
```

DWT Using Software

```c
void apply_filter(int* filtred_signal, int input_signal[],
  unsigned int input_length, int filter_bank[], unsigned int filter_length) {

  int* signal_window = (int*) 0x00100408;
  unsigned int edge_index;

  for(edge_index = 1; edge_index < (input_length+filter_length); edge_index = edge_index + 2) {
    signal_window[0] = input_signal[edge_index];
    signal_window[1] = input_signal[edge_index-1];
    int tmp = 0; unsigned int i;
    for(i = 0; i < filter_length; i = i + 1) {
      tmp  = tmp + signal_window[i] * filter_bank[i];
    }
    filtred_signal[edge_index] = tmp;
    for(i = filter_length - 1; i >= 1; i = i - 1) {
      signal_window[i] = signal_window[i-1];
    }
    for(i = filter_length - 1; i >= 1; i = i - 1) {
      signal_window[i] = signal_window[i-1];
    }
  }
  return;
}
```

```c
void wavelet_analysis(int* dwt_coeffs, int decomposition_level,
  unsigned int signal_length, unsigned int filter_length) {

  int* cur_inputs = &test_inputs[0];
  int* detail_coeffs = (int*) 0x00100200;
  int* approx_coeffs = (int*) 0x00100300;
  int dec_idx; unsigned int i; unsigned int coeffs_length; unsigned int prev_coeffs = 0;

  int cur_inputs_length = signal_length;
  for(dec_idx = 0; dec_idx < decomposition_level; dec_idx = dec_idx + 1) {

    apply_filter(dwt_coeffs, cur_inputs, cur_inputs_length, hid_coeffs, filter_length);
    coeffs_length = ((cur_inputs_length + filter_length - 1) >> 1);
    for(i = 0; i < coeffs_length; i = i + 1) {
      dwt_coeffs[prev_coeffs+i] = detail_coeffs[i];
    }

    apply_filter(approx_coeffs, cur_inputs, cur_inputs_length, lod_coeffs, filter_length);
    prev_coeffs = prev_coeffs + coeffs_length;
    for(i = 0; i < coeffs_length; i = i + 1) {
      cur_inputs[i] = approx_coeffs[i];
    }

    cur_inputs_length = coeffs_length;
  }

  for(i = 0; i < coeffs_length; i = i + 1) {
    dwt_coeffs[prev_coeffs+i] = approx_coeffs[i];
  }
}
```

Section 5

Conclusion

Looking Back …

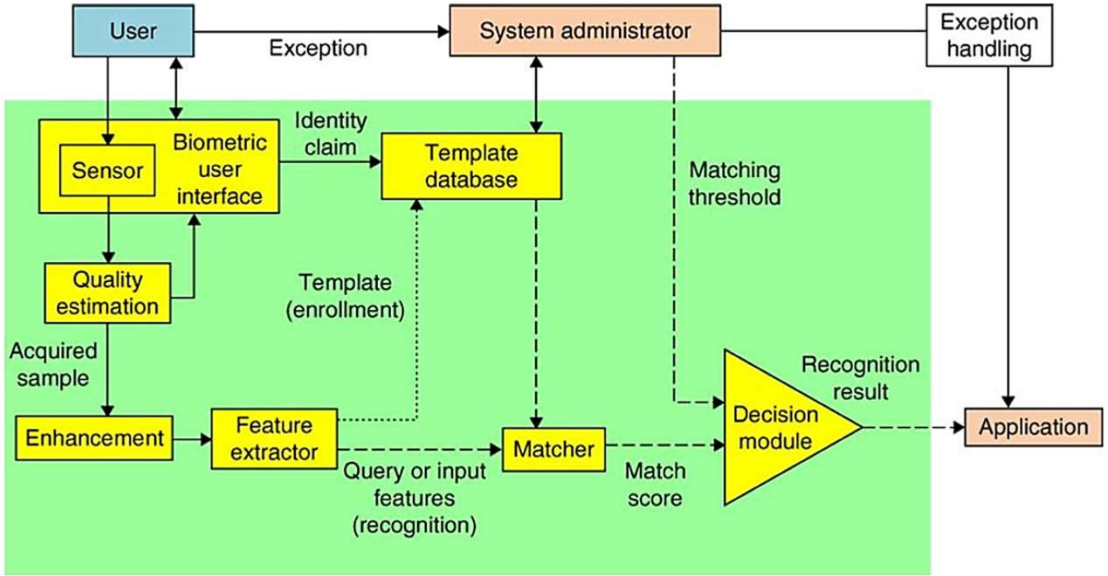
1.Signal Acquisition:
The first step involves the acquisition of signals from biological sources such as electrodes, sensors or stimulation devices.

2.Signal Pre-processing:
This stage involves removing noise, filtering, and amplifying the signals to improve the quality of the data.

3.Signal Analysis:
The processed signals are analyzed to extract relevant information, such as detecting specific patterns, calculating measurements or estimating parameters.

4.Signal Interpretation:
This stage involves the interpretation of the results obtained from the signal analysis, providing an understanding of the biological processes being monitored.

5.Signal Control and Actuation:
In some cases, the processed signals are used to control and actuate devices to influence biological processes.



What's been Achieved.

Mean Speed-up	Mean RMS Error
91.4	$2.56e^{-8}$

- ❑ An Inversible DWT
- ❑ Practical Alternative to Software-based DWT Calculation on AFTAB
- ❑ Platform for Future Optimization and Expansions
- ❑ Applicable For Different Use Cases