Vector/Array/Neuron Processor Design Plus Semester Highlights

Conor Csongradi ‘20, Lacie Flores ‘20, Ryan Schick ‘18

EGR 433: Advanced Computer Engineering

Elizabethtown College, Elizabethtown, PA

Processor Design Requirements:

Create 4 parallel scalar functional units with Vector Registers Vi, Vj, and Vk created from Ri’s, Rj’s and Rk’s of the 4 units. Use a 32-bit adder to add the two 16-bit product results from each unit. Put result into a 32 bit scalar accumulator, then into a neuron transfer function. Create an embedded code stack, controlled by a program counter via a master control unit with a Finite State Machine that implements the simple pipeline of Fetch, Decode, Execute, Write-Back, plus any special states. Embed a carefully crafted assembly language code segment to demonstrate the functionality of your instruction set.

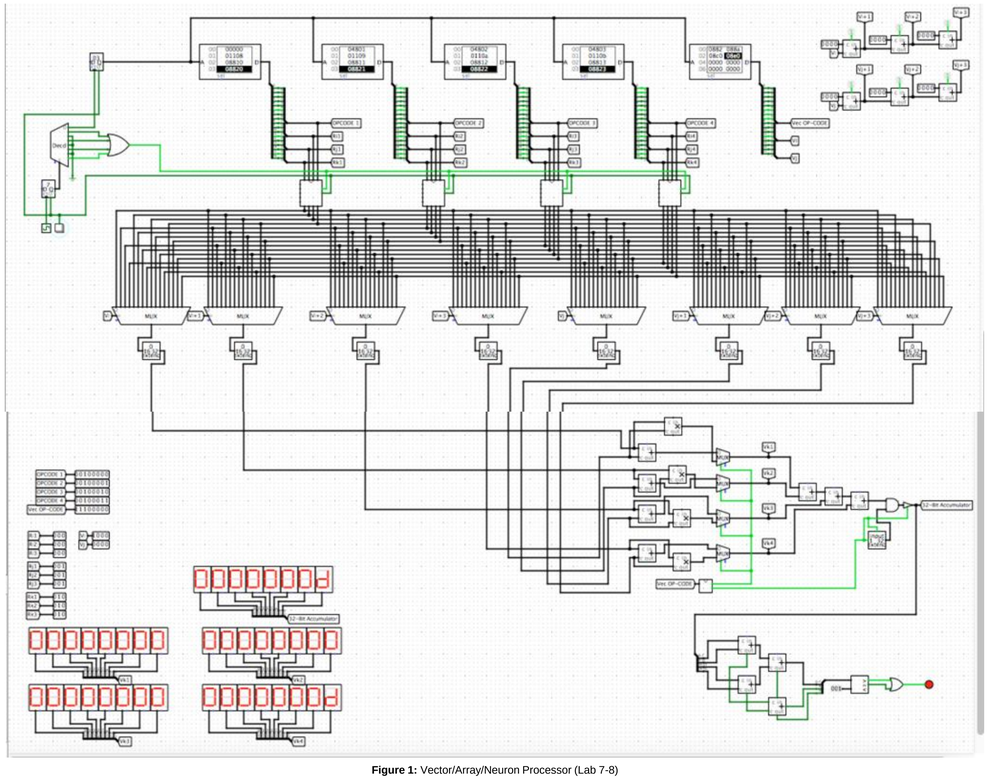

Figure 1: Vector/Array/Neuron Processor (Lab 7-8)

Super Scalar Design Highlight:

Two parallel pipelines in a 2-way super scalar architecture.Uses a finite state machine that alternates between pipes.

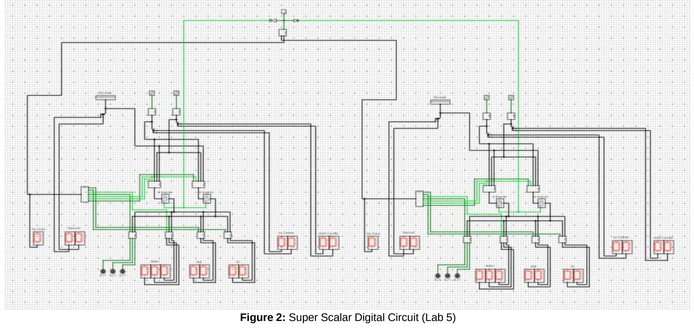

Figure 2: Super Scalar Digital Circuit (Lab 5)

Advanced PLC Highlight:

- Created Complex Voting Machine
  - Two Different Methods
    - Ladder Logic and Gates
      - Used in Industry, Aging Standard
    - Structured Text
      - High Level Programming Language
      - Global Variables Tied to I/O of Machine
  - LED’s Triggered Based on Voting Criteria of Four Different Groups
  - Output LED’s Also Triggered Based on Output Arithmetic
  - Versatility of $5,000 Advanced PLC Controller

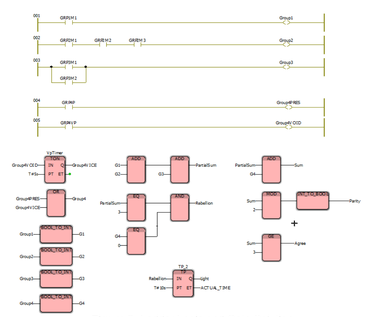

Figure 3: Ladder Logic and Gate Solution

```
(*  EGR433 Lab 7-8 PLC Program
    Ryan Schick, Conor Csongradi, Lacie Flores
                                                    *)

(*  First Group: One Member, What He Says Goes  *)
    If  GRP1M1
        Then    Group1 :=True;
        Else    Group1 :=False;
    End_If;

(*  Second Group: Three Members, All Must Agree  *)
    If  GRP2M1 & GRP2M2 & GRP2M3
        Then    Group2 :=True;
        Else    Group2 :=False;
    End_If;

(*  Third Group: Two Members, Either Can Agree  *)
    If  GRP3M1 OR GRP3M2
        Then    Group3 :=True;
        Else    Group3 :=False;
    End_If;

(*  Presidential Group: P and VP, Either Works, But VP on 5s Delay  *)

    (*  President  *)
    If  GRP4P
        Then    Group4 :=True;
        Else    Group4 :=False;
    End_If;

    (*  Vice President  *)
    If  GRP4VP & GRP4P =False
        Then    Timer1(IN :=True, PT :=T#5s); (* Starts Timer  *)
        Else    Timer1(IN :=False,PT :=T#5s);
    End_If;

    (*  VP Timer Condition  *)
    If  Timer1.Q =True OR GRP4P
        Then    Group4 :=True;
        Else    Group4 :=False;
    End_If;
```

Figure 4: Structured Text Solution Excerpt

NanoLC Highlight:

- Implemented 2-Bit Up and Down Counters
  - Cycles Through all Possible Steps/Iterations
- Stores Counter Output in Registers
  - Comparator Designed Around Register Arithmetic
- Counter and Comparator Values Displayed via LED’s
- Programmed via Flowchart Logic

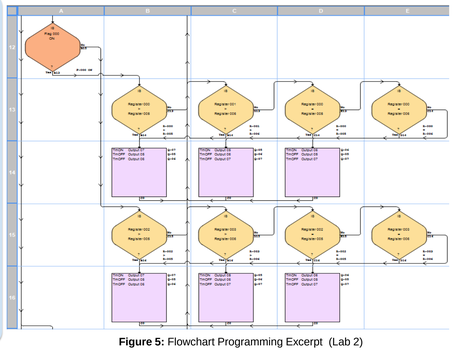

Figure 5: Flowchart Programming Excerpt (Lab 2)

FPGA/Xilinx Highlight:

- Created Digital Circuit Controlled By Four Bit Instruction Set
- Capable of 8-Bit Arithmetic
  - Stores Results in Registers
  - Outputs to Hex Display
- All 8-Bit Functional Blocks Created From Scratch

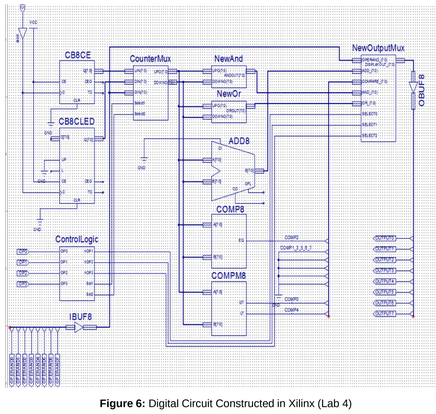

Figure 6: Digital Circuit Constructed in Xilinx (Lab 4)

Breadboarding/Power Supply Highlight:

- Modified an Old PSU to Serve as Power Supply for Breadboard Test Bench
- Stripped and Organized Wires According to ATX Layout
- Interfaced Testing Bench with NanoLC

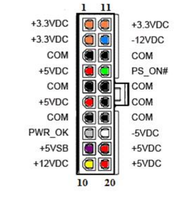

Figure 7: 20 Pin ATX PSU Pinout (Lab 4)