GALOIS FIELD ARITHMETIC UNIT

Sabbir Ahmed, Jeffrey Osazuwa, Howard To, Brian Weber

CMPE 451: Critical Design Review

Mission Statement

- Complete a scalable design of a Galois Field Arithmetic Unit capable of generating Galois fields and executing operations within the generated field.
- Emphasis on design

Overview

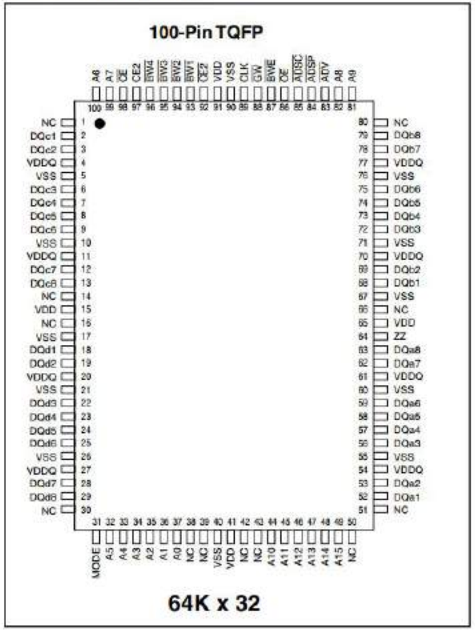
SRR	PDR	CDR
Introduction to Concepts	Hardware Configuration Items	Electrical Design
Functional Flow	Software Configuration Items	Mechanical Design
Data Flow	Interface Requirements	Memory Architecture
Trade Studies		Input/Output
Testing Methods		Modules and Libraries
		Final Demo

Progress Since PDR

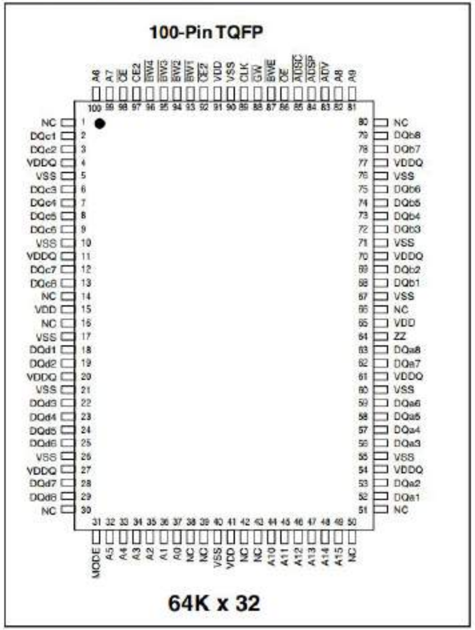
- Finished most VHDL modules
    - Experimented with different design philosophies
- Purchased all necessary hardware
    - Soldered pins
- Finished designing I/O handler
- Started interfacing FPGA with memory
    - Improved memory architecture
    - Started testing with memory

Changes in Memory

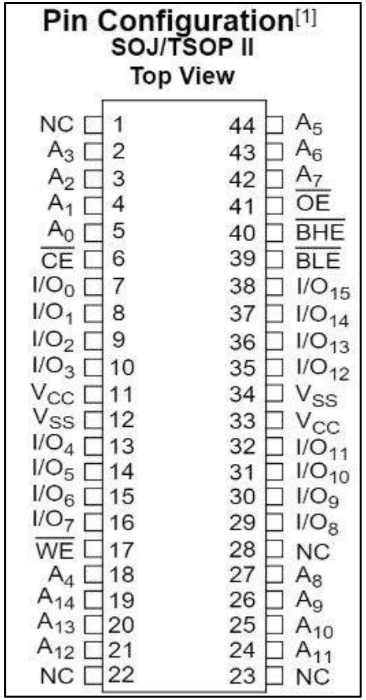
PIN CONFIGURATION



PIN CONFIGURATION

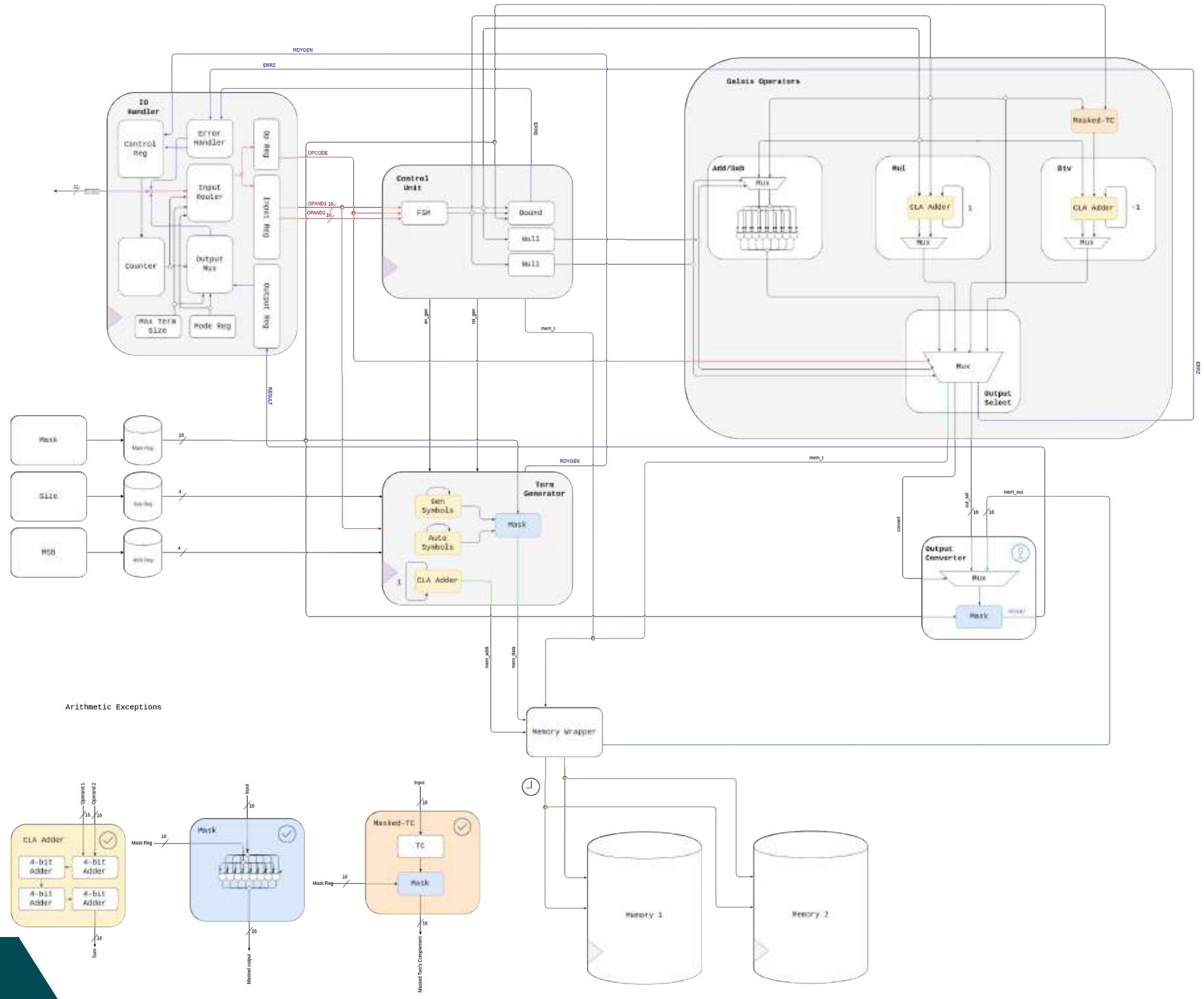


64K x 32 Synchronous Pipelined SRAM

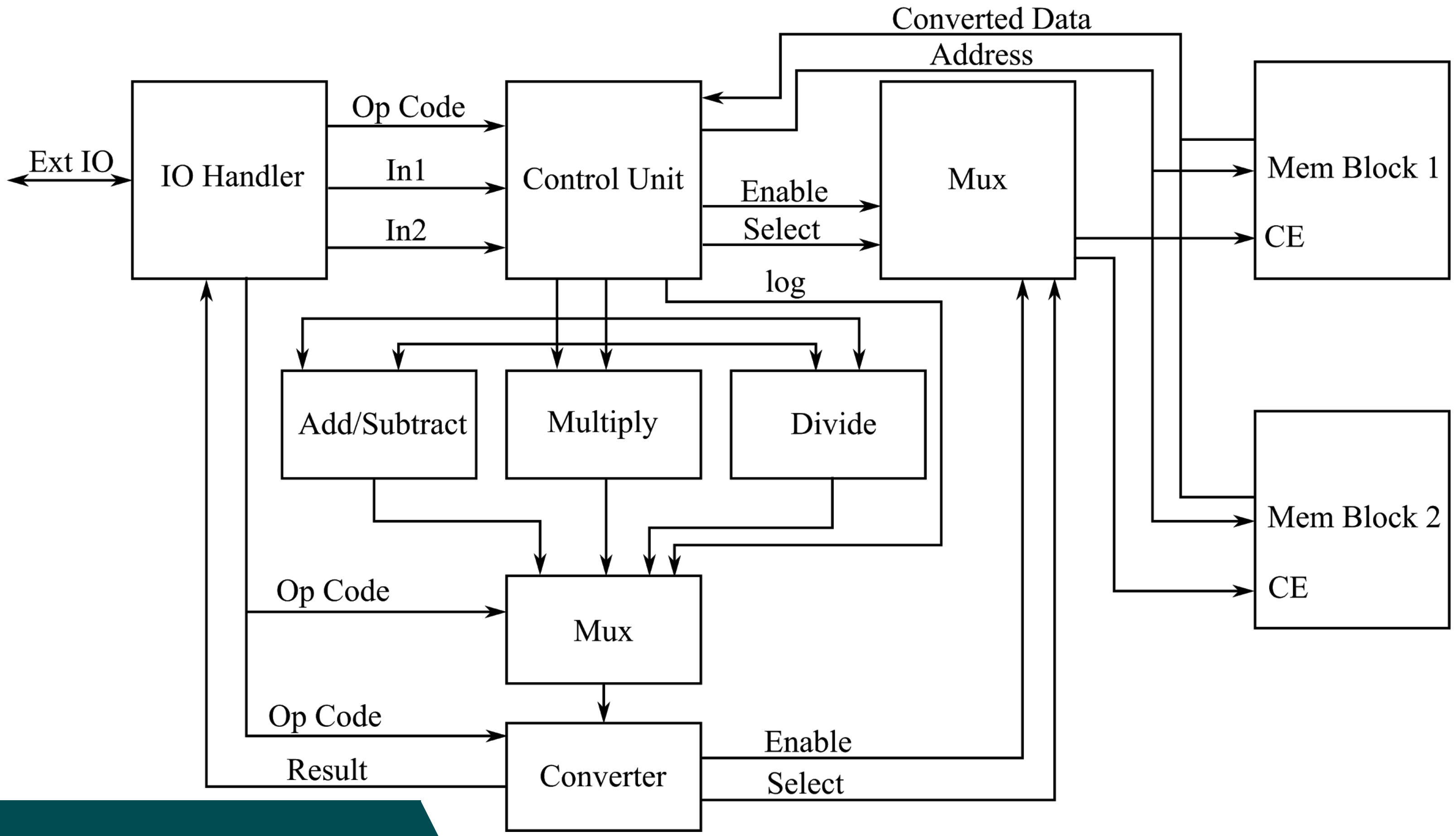

Pin Configuration[1]
SOJ/TSOP II
Top View

32K x 16 Asynchronous SRAM

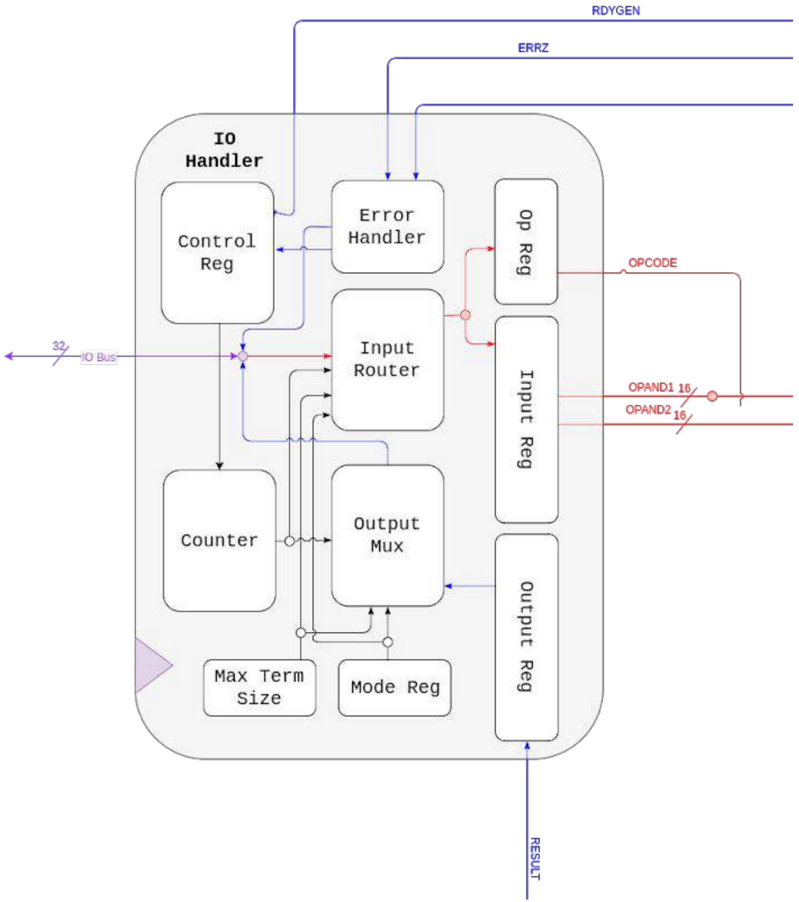
HARDWARE CONFIGURATION ITEMS



Block Schematic

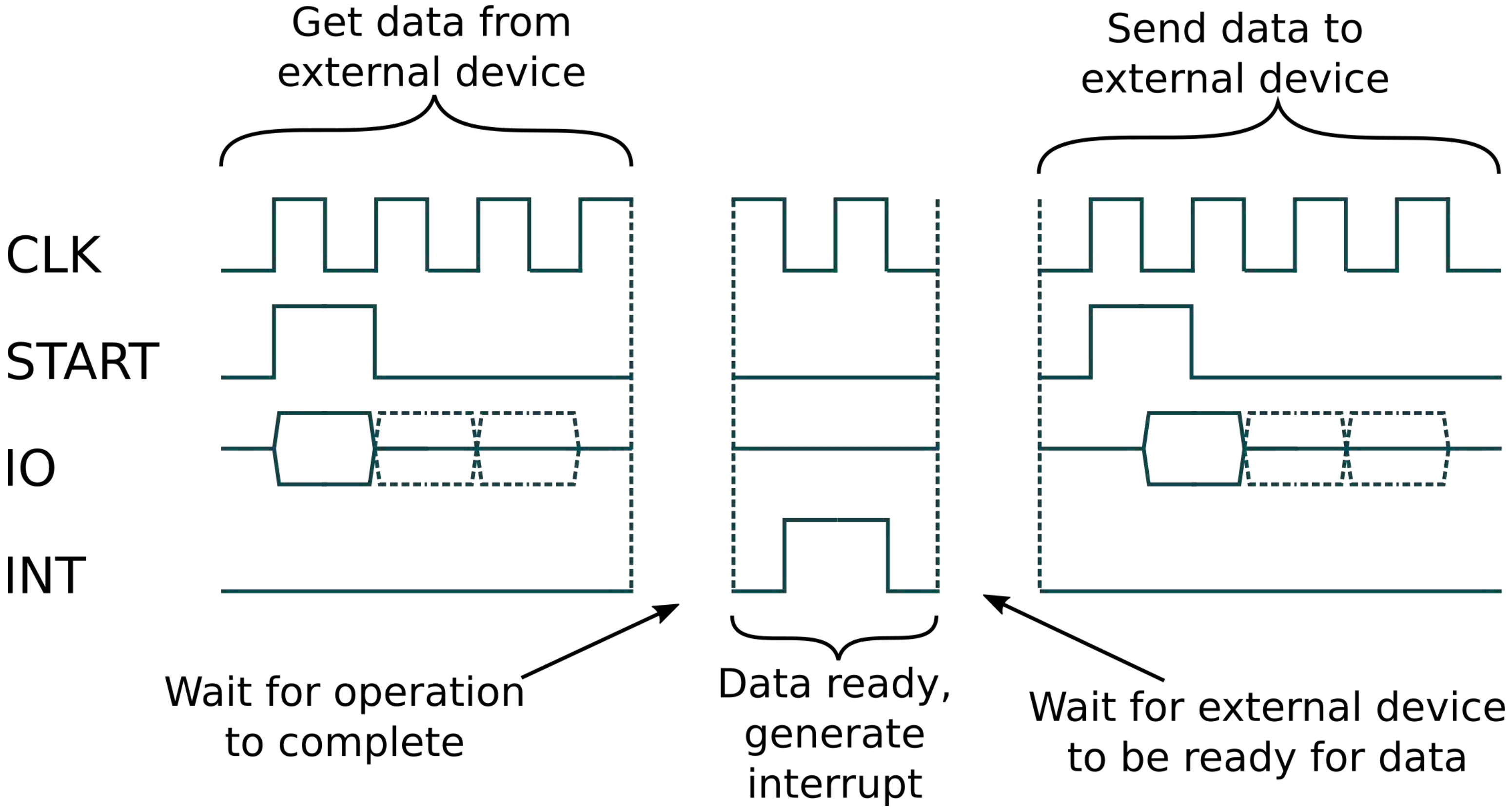


High Level System View


RDYGEN
ERRZ
IO Handler
Control Reg
Error Handler
Op Reg
OPCODE
32
IO Bus
Input Router
Input Reg
OPAND1 16
OPAND2 16
Counter
Output Mux
Output Reg
Max Term Size
Mode Reg
RESULT

I/O Handler

Timing Diagram


Get data from external device
Send data to external device
CLK
START
IO
INT
Wait for operation to complete
Data ready, generate interrupt
Wait for external device to be ready for data

Input / Output

- Relatively standard protocol
- All handled by the I/O Handler Module
- Design completed since PDR
- Working on getting design to synthesize
- Will write C libraries/drivers to handle protocol

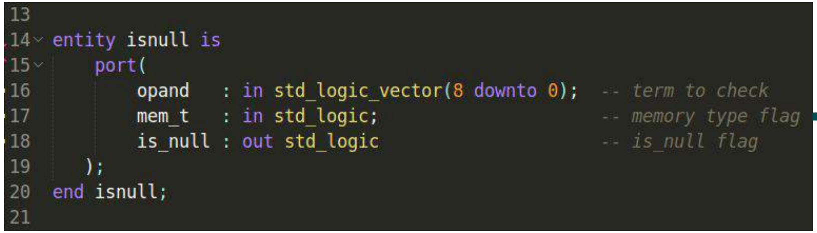
SOFTWARE CONFIGURATION ITEMS

Scalability: *n*-degree Polynomials

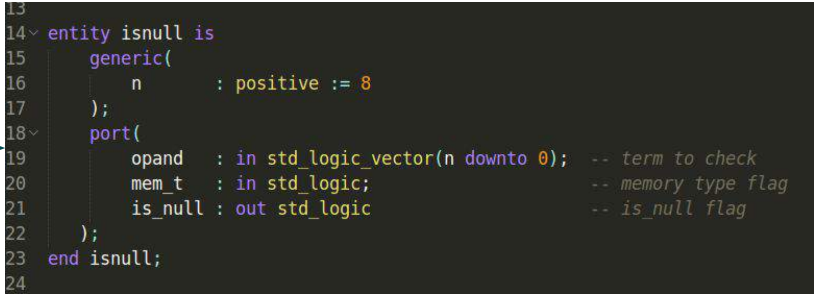
- New memory architecture now allows *n*-degree polynomials
  - where n = # of address pins - 1 ≤ # of data pins - 1
- Parameterized modules
  - Allows synthesis of $(n\text{-}m)$-degree polynomials
    - where $0 \leq m < n$
- Minimizes unnecessary hardware

Scalability: Parameterized Modules

- VHDL Generics



```
entity isnull is
    port(
        opand   : in std_logic_vector(8 downto 0);  -- term to check
        mem_t   : in std_logic;                     -- memory type flag
        is_null : out std_logic                     -- is_null flag
    );
end isnull;
```



```
entity isnull is
    generic(
        n       : positive := 8
    );
    port(
        opand   : in std_logic_vector(n downto 0);  -- term to check
        mem_t   : in std_logic;                     -- memory type flag
        is_null : out std_logic                     -- is_null flag
    );
end isnull;
```

- Python helper scripts
    - Dynamically generate code for fixed positioned priority encoders

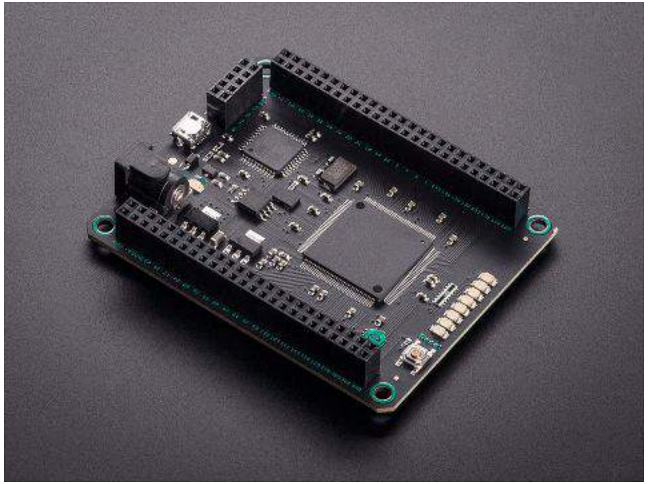
External C Libraries

- I/O driver
- `primgen` library
  - Wraps `primpoly`
    - open-source primitive polynomial generation program
  - Checks primitivity of inputted polynomials

FINAL DEMO

Demo FPGA

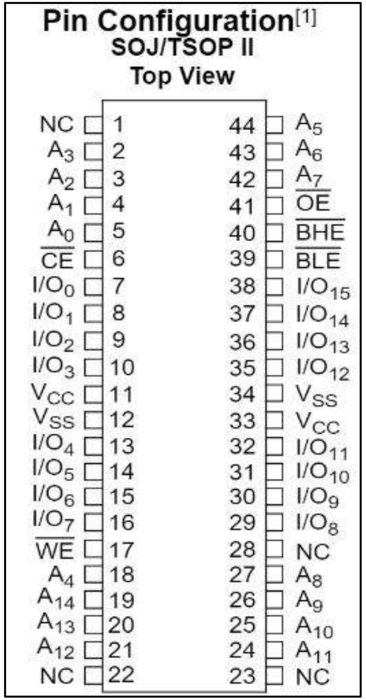
Mojo FPGA Development Board



Source: adafruit

Demo Memory Specifications

- 10 ns write and access times
- 32 KB of memory in 16 bit words
- 15 address pins
- 16 data pins
- 5 control Pins



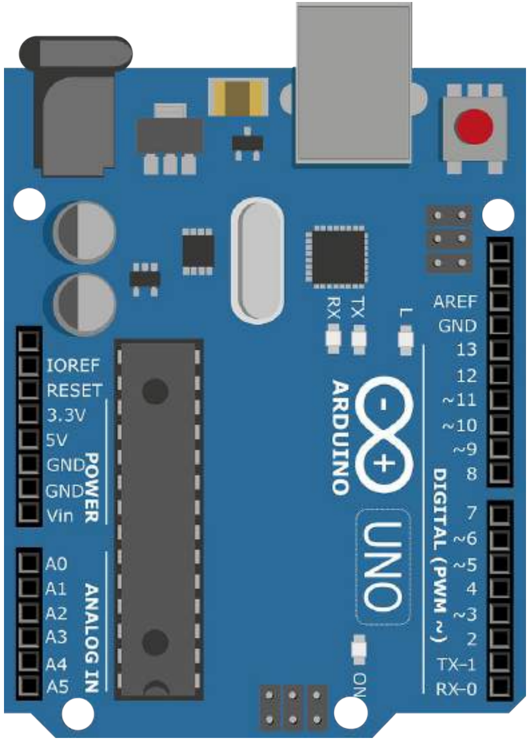
Pin Configuration[1]
SOJ/TSOP II
Top View

NC	1	44	A_5
A_3	2	43	A_6
A_2	3	42	A_7
A_1	4	41	$\overline{OE}$
A_0	5	40	$\overline{BHE}$
$\overline{CE}$	6	39	$\overline{BLE}$
I/O_0	7	38	I/O_{15}
I/O_1	8	37	I/O_{14}
I/O_2	9	36	I/O_{13}
I/O_3	10	35	I/O_{12}
V_{CC}	11	34	V_{SS}
V_{SS}	12	33	V_{CC}
I/O_4	13	32	I/O_{11}
I/O_5	14	31	I/O_{10}
I/O_6	15	30	I/O_9
I/O_7	16	29	I/O_8
$\overline{WE}$	17	28	NC
A_4	18	27	A_8
A_{14}	19	26	A_9
A_{13}	20	25	A_{10}
A_{12}	21	24	A_{11}
NC	22	23	NC

32K x 16 Asynchronous Pipelined Static RAM

Demo I/O Specifications

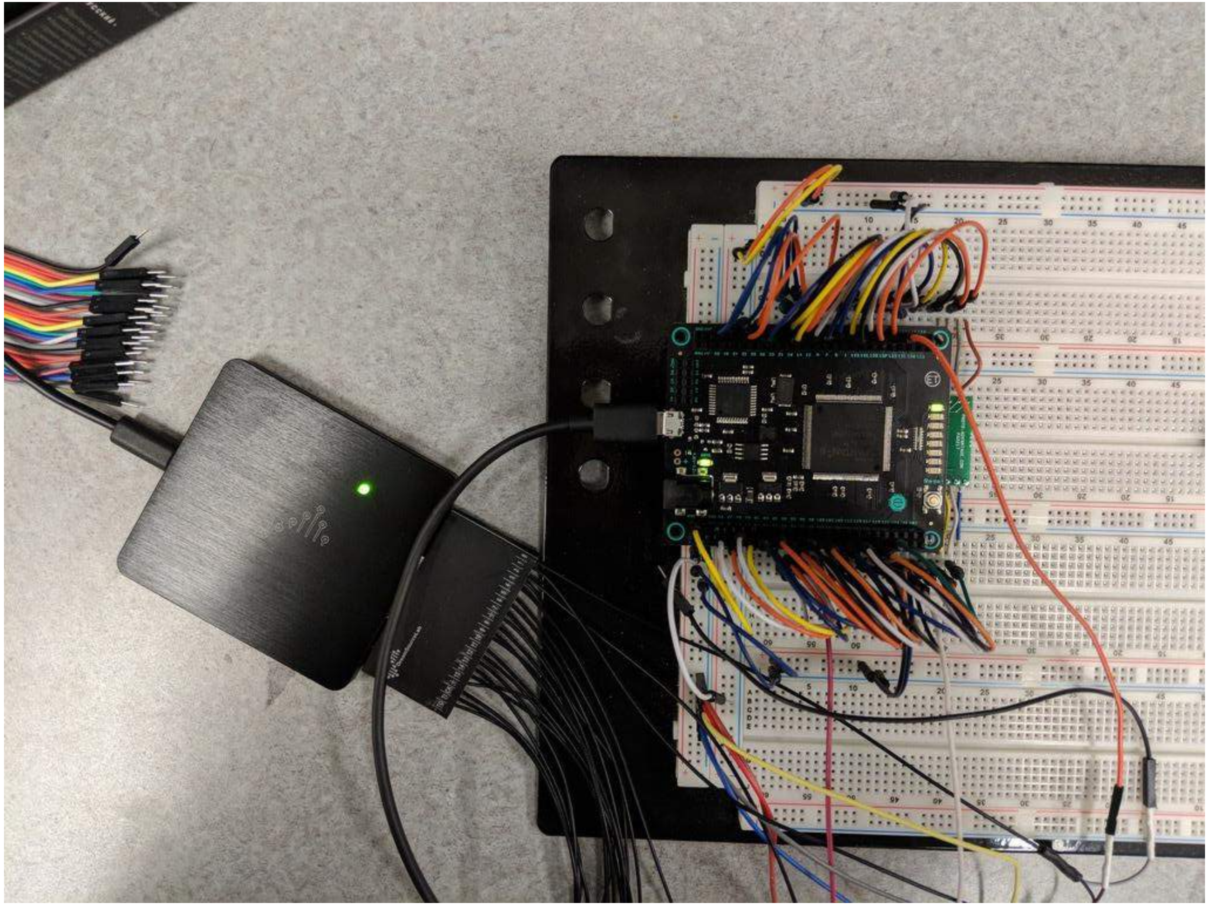
- Operating Voltage: 5 V
- Clock speed: 16 MHz
- SRAM: 2 KB
- EEPROM: 1 KB
- Flash Memory: 32 KB



Arduino Uno Rev3

Demo ALU Specifications

- Handle 8th degree polynomial
- Generate all GF(2^8)terms
- Demonstrate all operations
- Generate arithmetic exceptions
  - Divide-by-zero exception
  - Log-of-zero exception
  - Upper-bound exception



GFAU Components

Schedule

- Finalize all VHDL modules over spring break
- Memory Interface
- I/O Interface
- Testing
- Demo

QUESTIONS?