BIT BANGED I²C REAL-TIME CLOCK COMMUNICATION

Beau Coburn and Aaron Foster

OVERVIEW

Interface with a Real Time Clock using I^2C Bit Banged on MSP430

Goal:

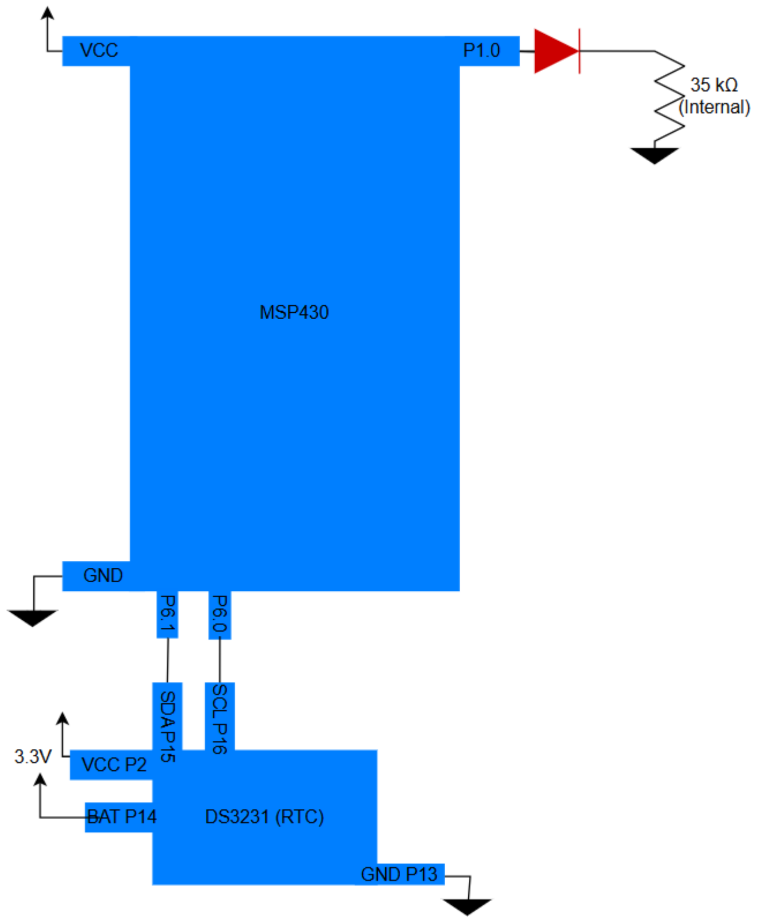
-Write multiple bytes to the RTC

-Read multiple bytes from RTC

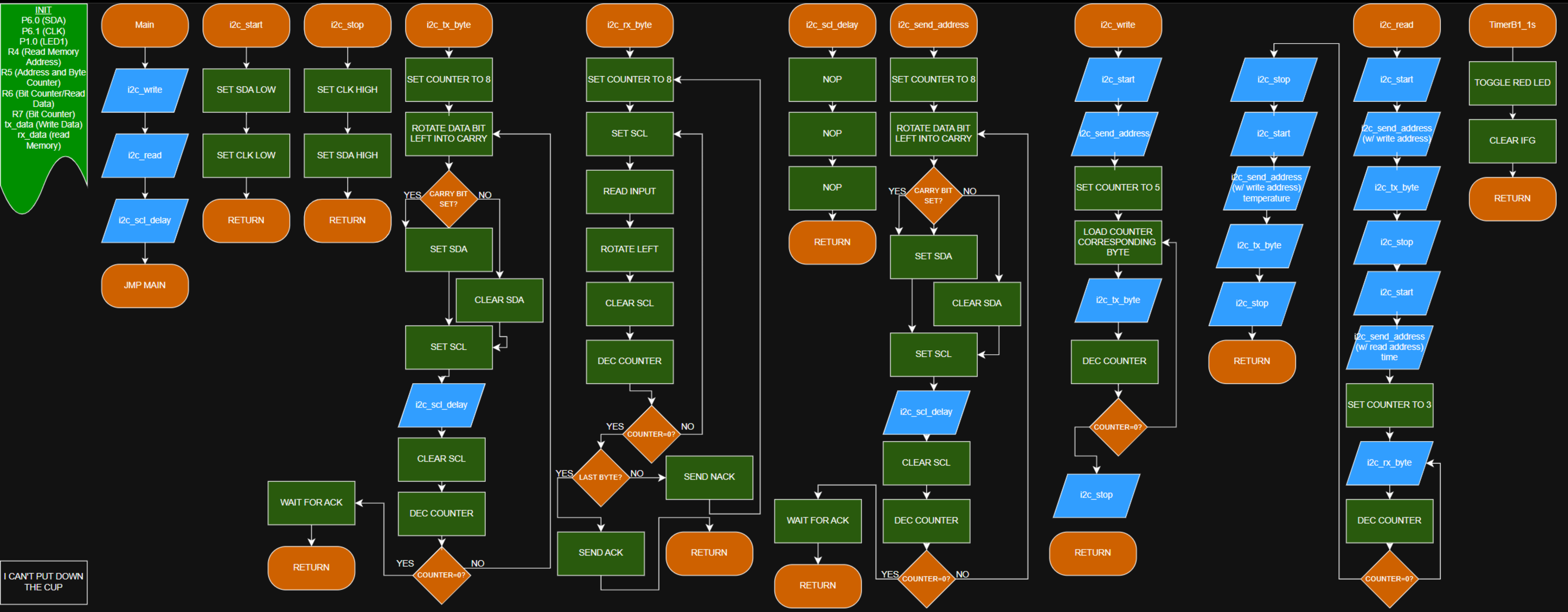
-Read the hours, minutes, seconds, and temperature registers

- Verify Time in counting up
- Interpret Temperature to be accurate

CIRCUIT DIAGRAM

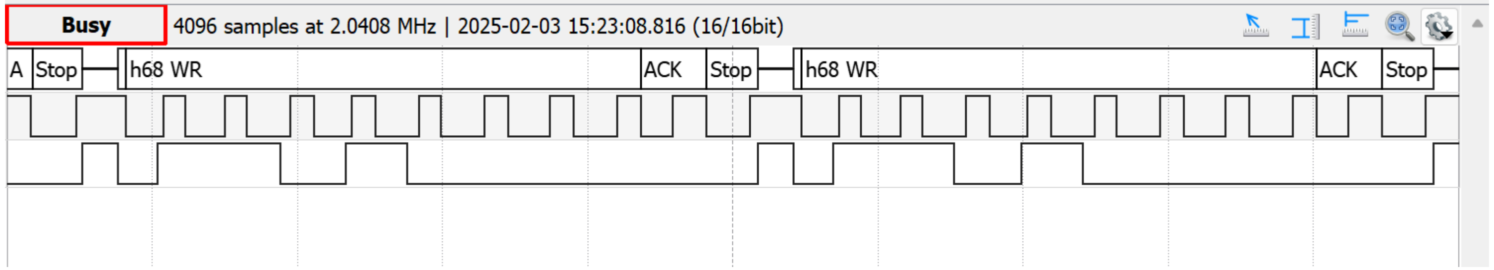


CODE FLOW AT A GLANCE



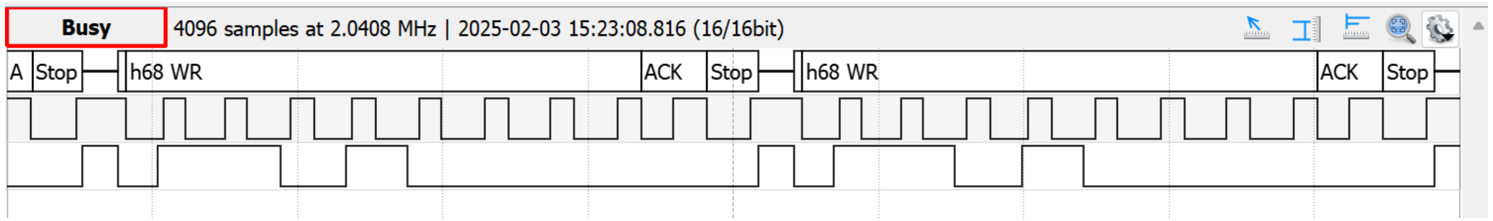
CODE DEVELOPMENT (START, STOP)

- Start Condition (Begins Every Transaction)
  - SDA Pulled Low While SCL High
- Stop Condition (Ends Every Transaction
  - SDA Pulled High While SCL High



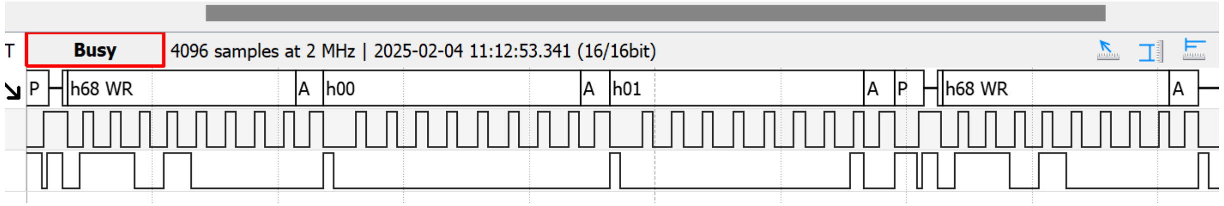
CODE DEVELOPMENT (ACKNOWLEDGE AND WRITE)

- Slave Address
  - RTC being used has an address of 68h
  - D0h used so that when rotated the write bit is set
  - Slave expecting data
- Acknowledge
  - Sent from slave recognizing a byte has been written



CODE DEVELOPMENT (SENDING BYTES)

- Sending Multiple Bytes
  - Each byte is rotated through in binary
  - Clock toggled for each bit
  - SDA pulled high or low depending on value
- Acknowledge sent from slave after each byte
- Transaction ended with a stop




```
Start, hD0 [ h68 | WR ], h00, h01, Stop
Start, hD0 [ h68 | WR ], h00, h01, Stop
```

CODE DEVELOPMENT (READING BYTES)

- Reading From the RTC
  - Send slave address with read bit (D1h)
  - Set SDA as input
  - Allow slave to send data to pin
  - Toggle SCL per bit of data sent
  - Send Acknowledge to Slave after each byte
    - Make SDA output to do s0
  - Repeat for number of expected bytes
  - Transaction Ends with a Nack

```
Start, hD1 [ h68 | RD ], h00, h01, h01 NACK, Stop
Start, hD1 [ h68 | RD ], h01, h00, h00 NACK, Stop
Start, hD1 [ h68 | RD ], h00, h00, h00 NACK, Stop
Start, hD1 [ h68 | RD ], h00, h00, h02 NACK, Stop
Start, hD1 [ h68 | RD ], h1C, h88, h00 NACK, Stop
Start, hD1 [ h68 | RD ], h15, h40, h03 NACK, Stop
```

CODE DEVELOPMENT (READING REGISTERS)

- Send Slave Address with write bit and register address to begin read
  - Send repeated start
- Send Slave address with read bit
  - Set number of bytes to be read
  - Send Ack after every byte read
  - End with Nack
  - Store read values in allocated memory

```
Start, hD1 [ h68 | RD ], h49, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h49, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h49, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h50, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h50, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h50, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h50, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h50, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h50, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h51, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h51, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h51, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h51, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h51, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h52, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h52, h04, h11, h02, h3F NACK, Stop
Start, hD0 [ h68 | WR ], h00, Stop
Start, hD1 [ h68 | RD ], h52, h04, h11, h02, h3F NACK, Stop
```

CODE DEVELOPMENT (READING TIME AND TEMP)

- Sent Write/Read transaction to read time registers
  - Registers are (00h, 01h, 02h)
  - Terminate Read
- Sent Write/Read transaction to read temperature registers
  - Registers are (11h, 12h)
  - 11h MSB (integer)
  - 12h LSB (decimal)
  - Celsius = MSB + (LSB/256)
- This loop repeats until program is terminated

END RESULT

- RTC can be seen initially being written to
- Following Read time registers can be observed counting up
- When RTC is warmed and cooled the temperature registers will update
  - Updates every 64 seconds