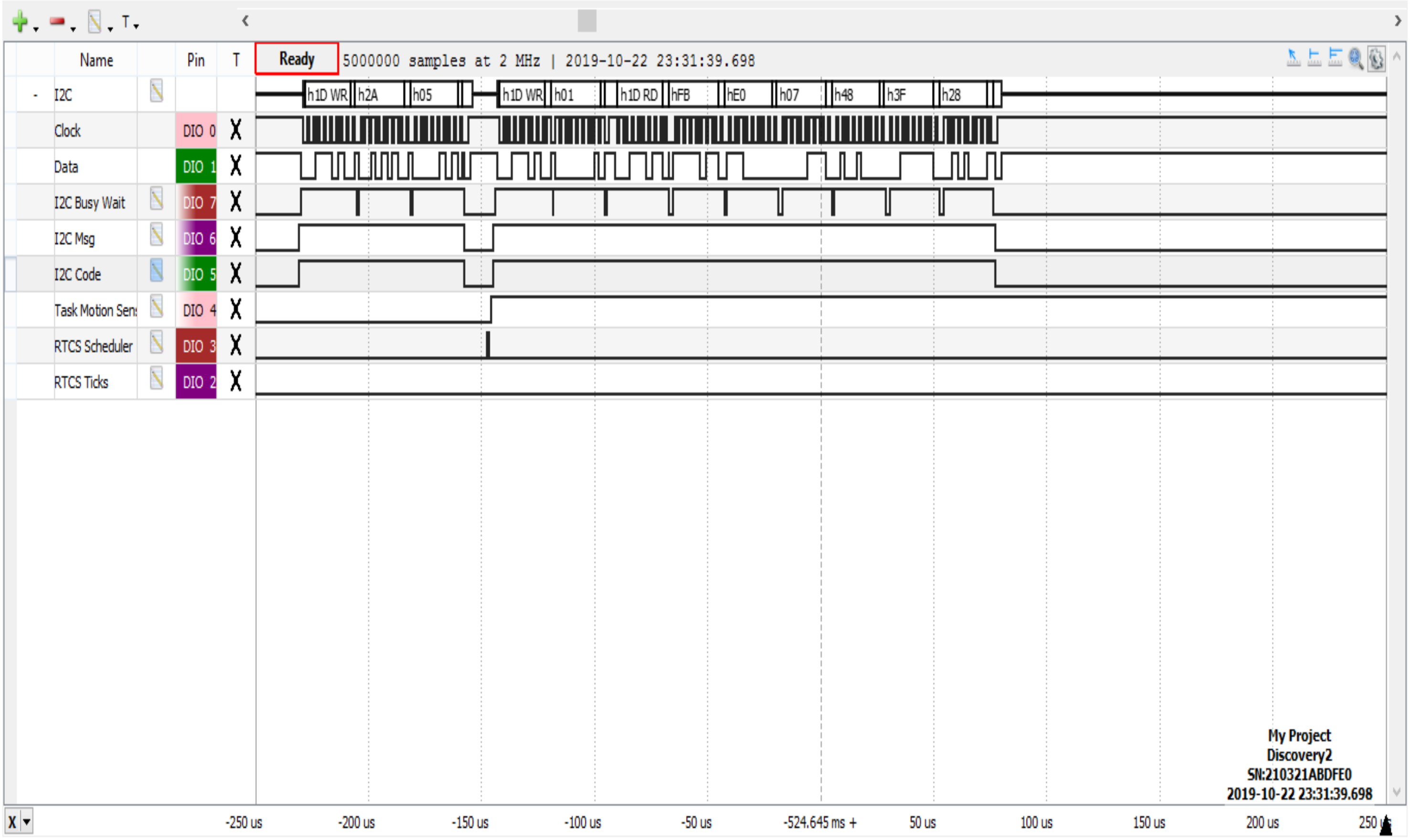
1. I2C Read – Blocking Mode



5000000 samples at 2 MHz | 2019-10-22 23:31:39.698

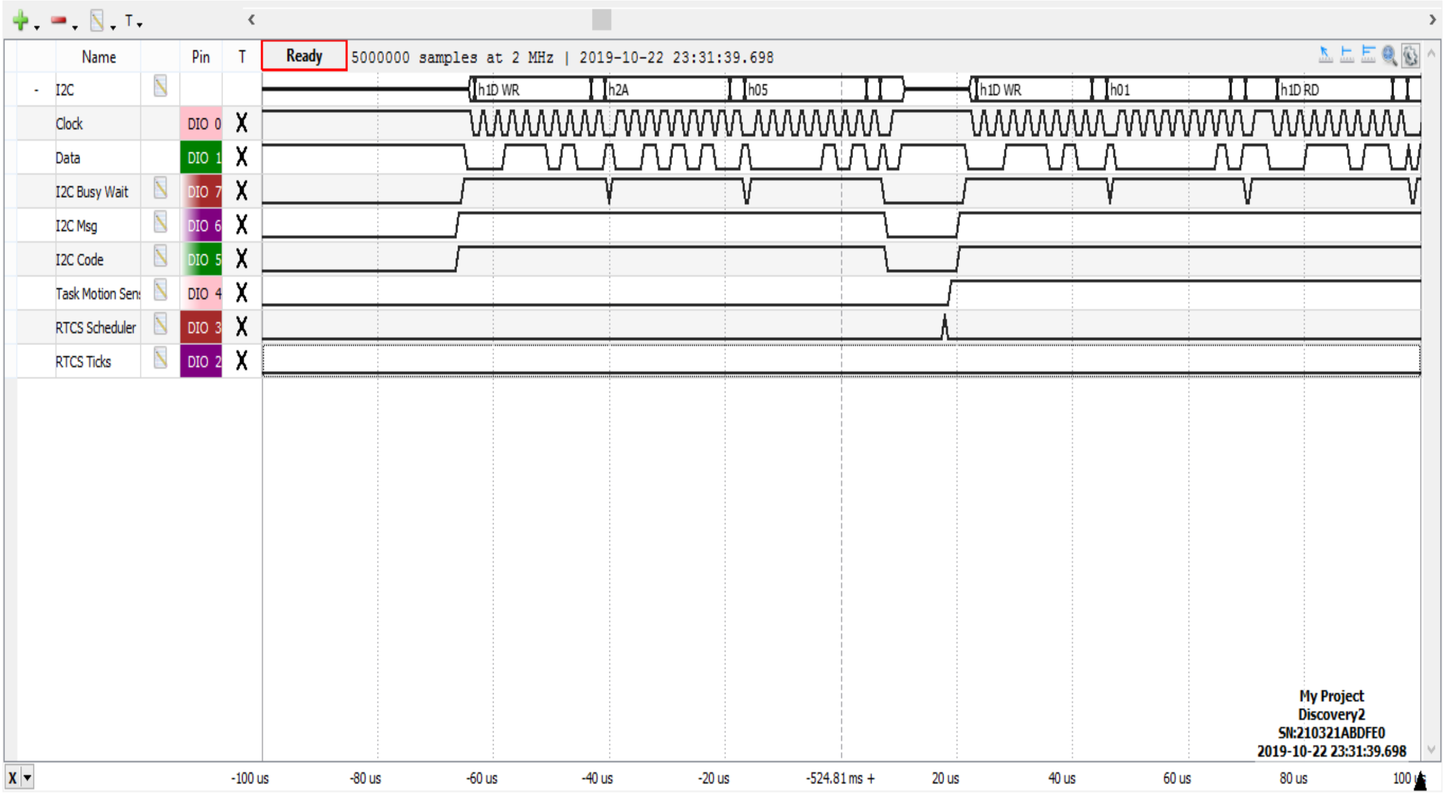
Task Motion Sensor – 12.76 ms

Task Motion Sensor Frequency - 50.499 Hz, runs every 19.83 ms

Message on Bus signal – 222 us

I2C Busy Wait – 220.5 us

2. I2C Write - Blocking



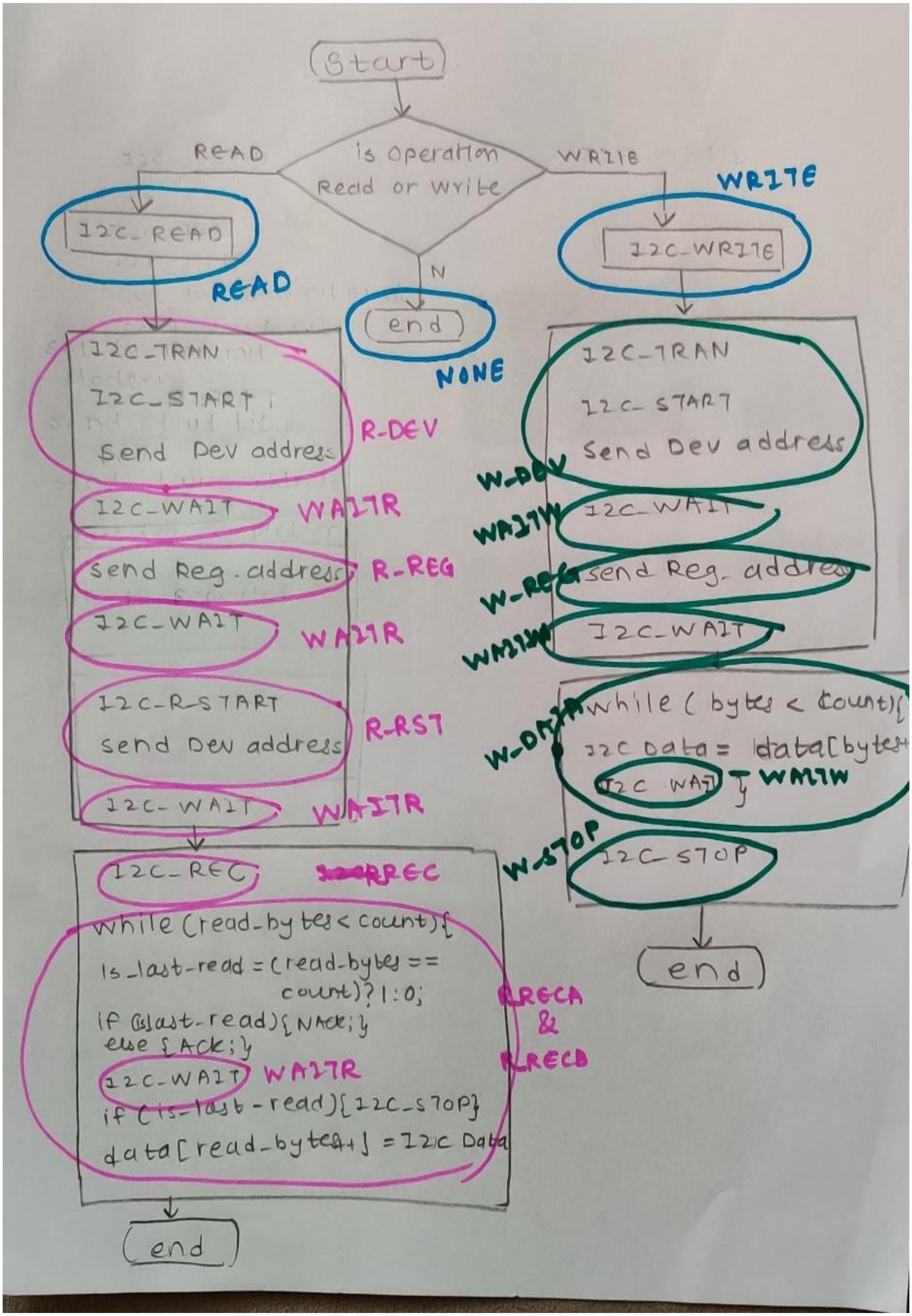
Ready

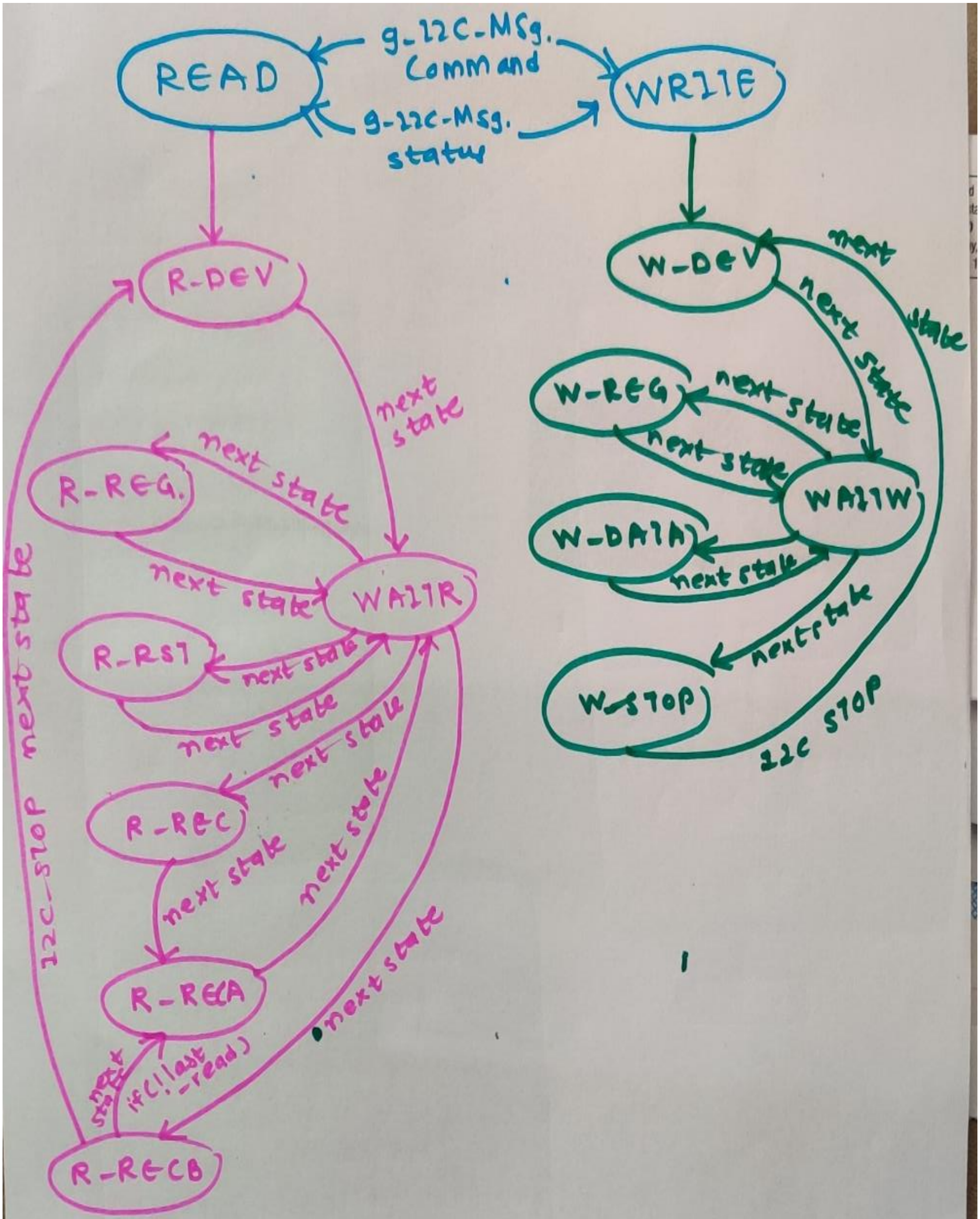
5000000 samples at 2 MHz | 2019-10-22 23:31:39.698

Message on Bus Signal – 74 us

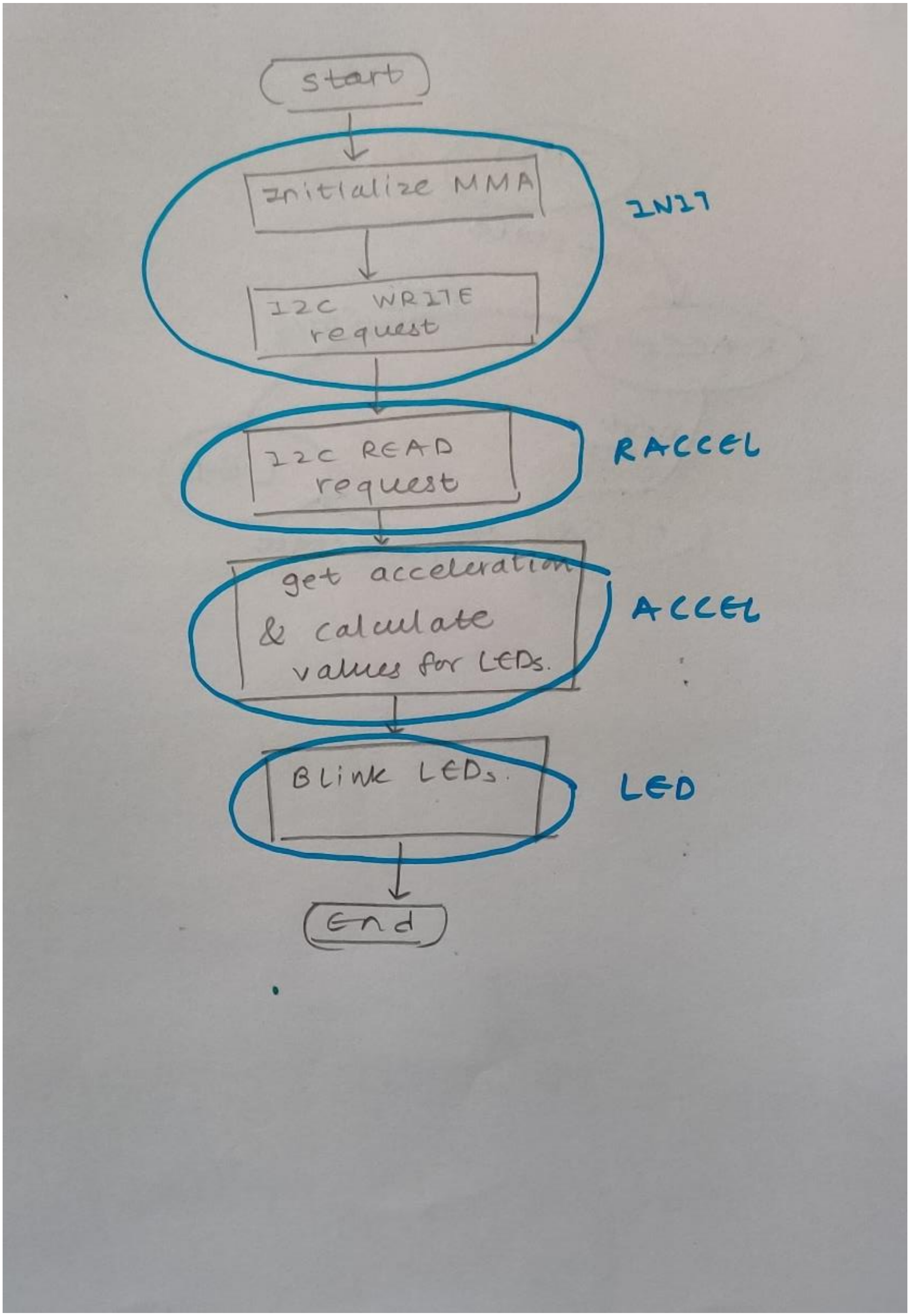
I2C Busy Wait – 72.5 us

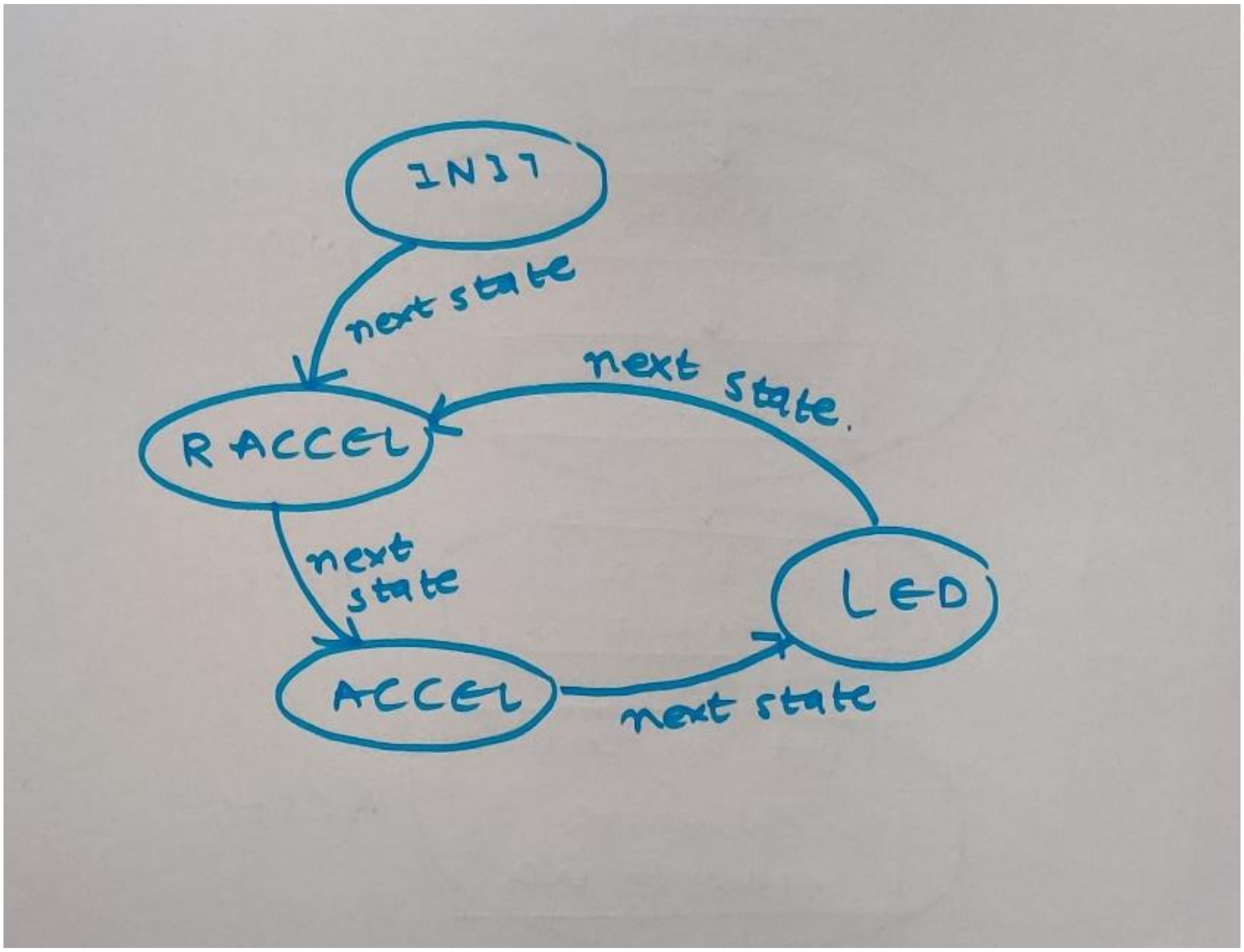
3. Task I2C Server FSM



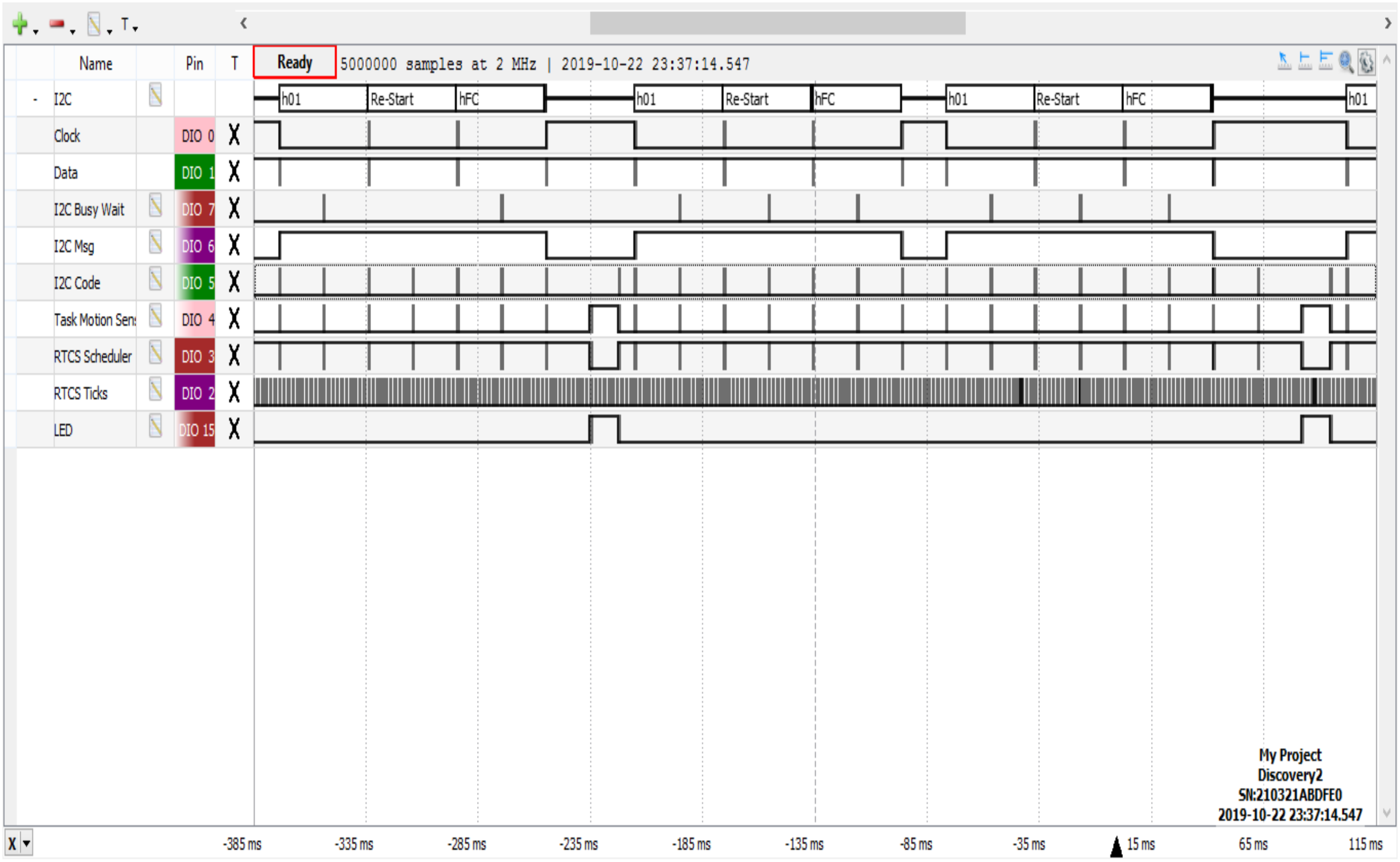

READ
g_I2C_Msg. Command
WRITE
g-I2C-Msg. status
R-DEV
next state
R-REG.
next state
next state
WAITR
R-RST
next state
next state
R-REC
next state
next state
next state
R-RECA
if(!last_read)
next state
next state
R-RECB
I2C_STOP next state
W-DEV
next state
next state
W-REG
next state
next state
WAITW
W-DATA
next state
next state
W-STOP
I2C STOP

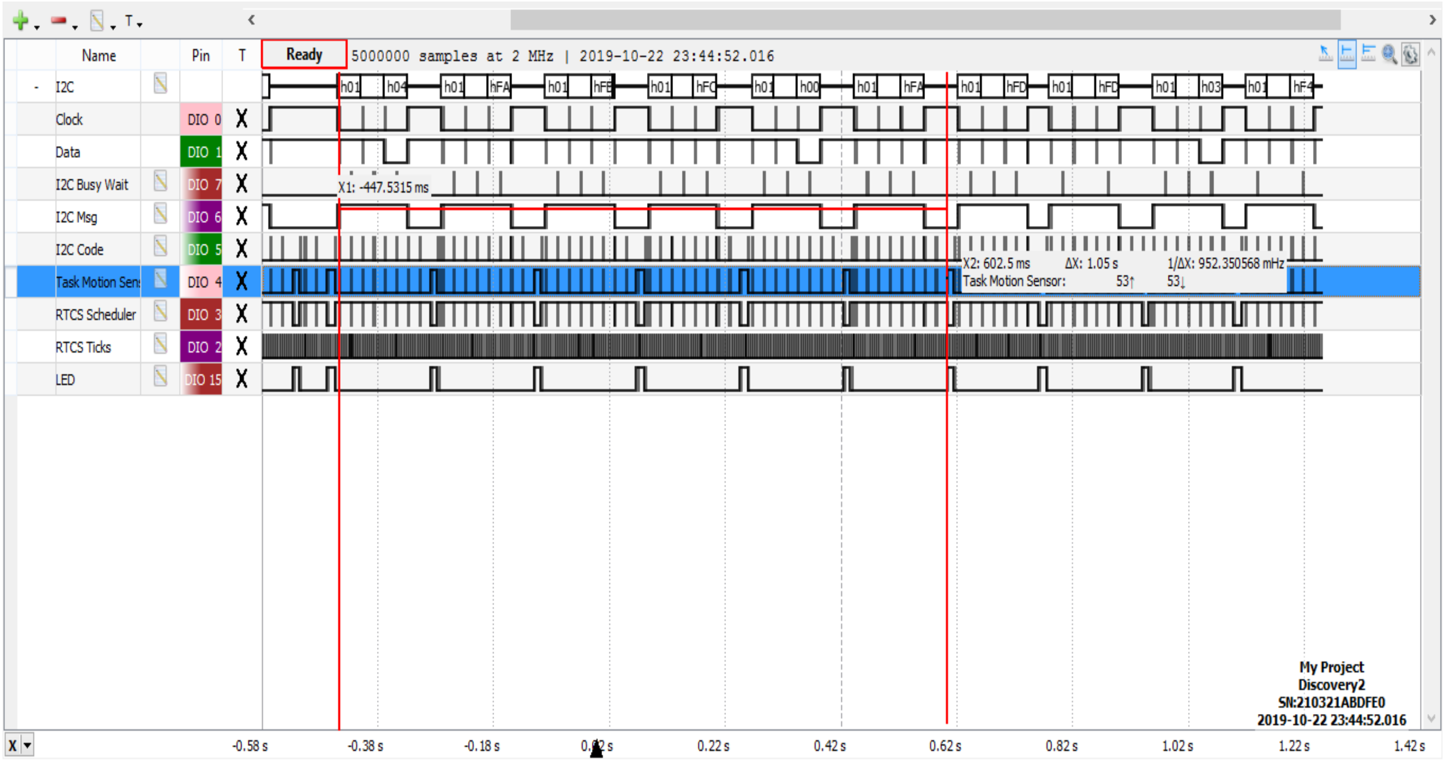
4. Task Motion Sensor FSM




INIT
next state
RACCEL
next state.
next state
LED
ACCEL
next state

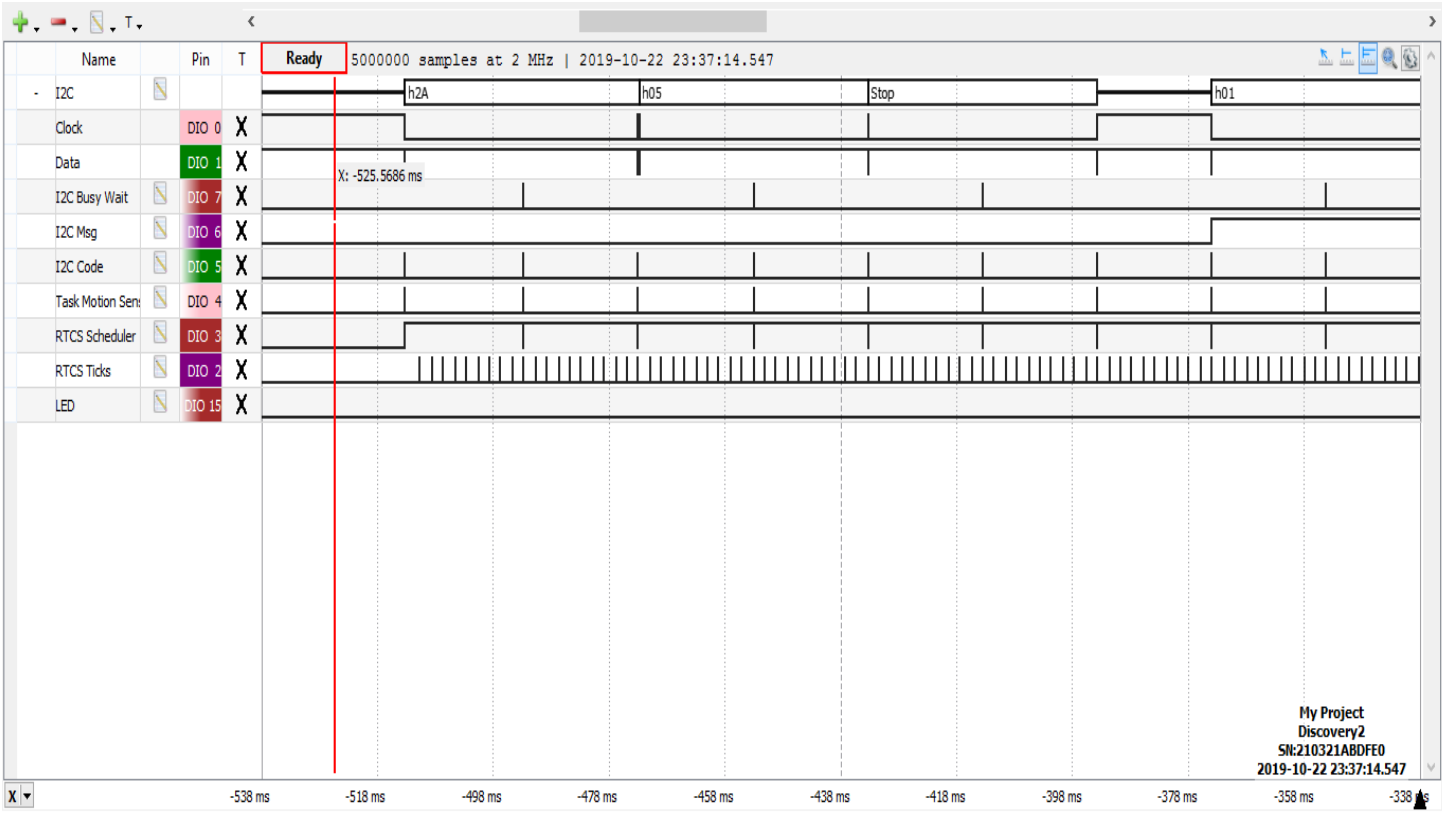
5. I2C Read – FSM Mode





Here, the time between sending I2C read request and lighting of LED by task motion sensor is 695.9 ms i.e. between those 2 red lines, as we are reading 6 bytes.

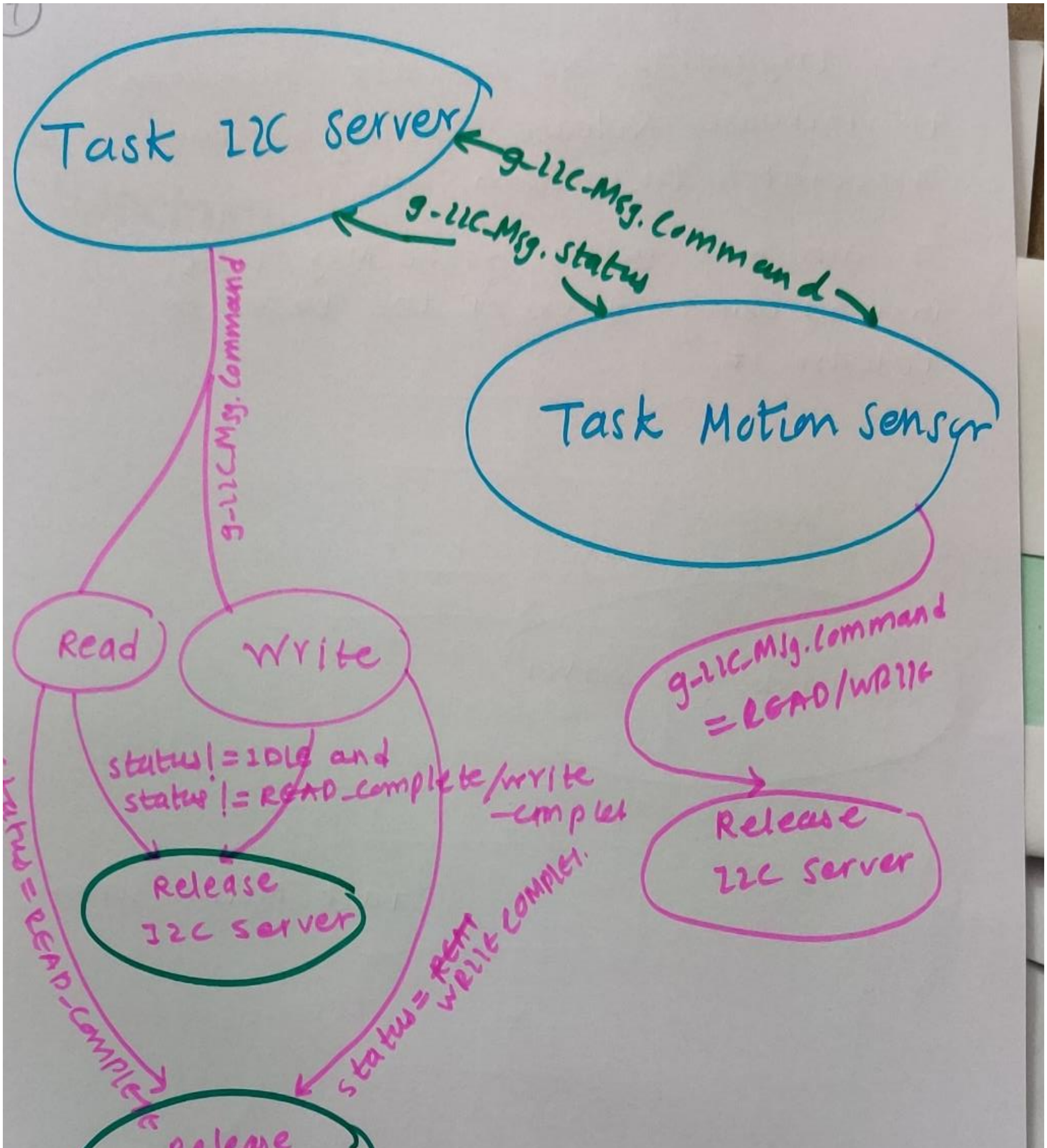
6.



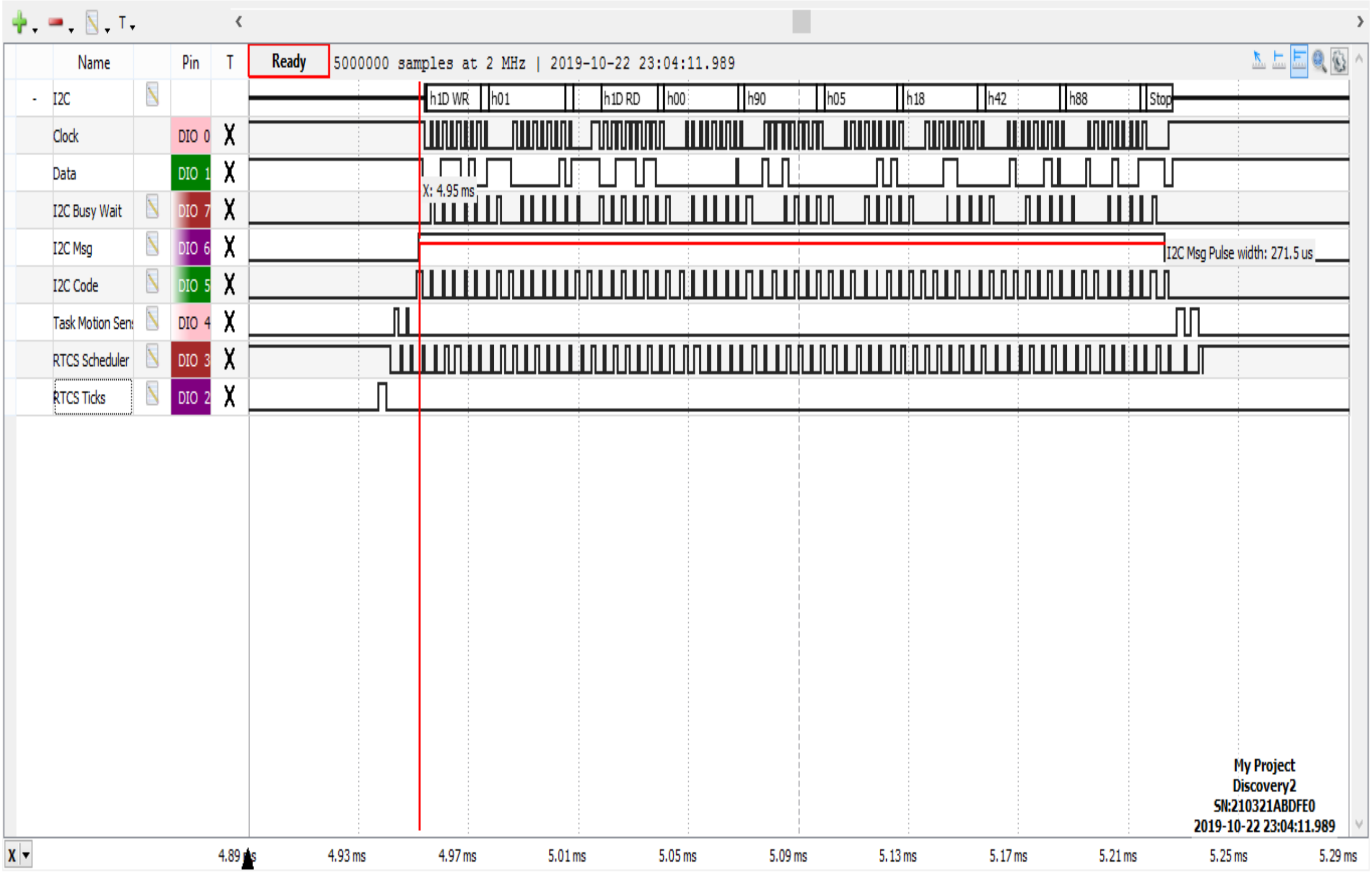
Ready 5000000 samples at 2 MHz | 2019-10-22 23:37:14.547

7. For event trigerring we can use function from RTCS API to increase Release Pending of the associated task. I am not using g_I2C_Msg.Client, instead using the priority index of the task to release the tasks. Overall approach is similar to the one asked in the project.

   For LED blinking, delay function typically takes 4 ms to execute. Now we will use scheduler which has timer with 2 ms for each ticks. Hence we will set the period of motion sensor FSM to 2 timer ticks for ON time and 4 timer ticks for OFF time, with releasing request for I2C Server.


Task I2C server
g-I2C-Msg.command
g-I2C-Msg.status
Task Motion Sensor
g-I2C-Msg.command
Read
Write
status != IDLE and status != READ_complete/write_complet
Release I2C server
status=READ_Complete
status= ~~READ~~ WRITE complet.
Release Motion sensor
g-I2C-Msg.command =READ/WRITE
Release I2C Server

8. I2C Read – Event Triggering

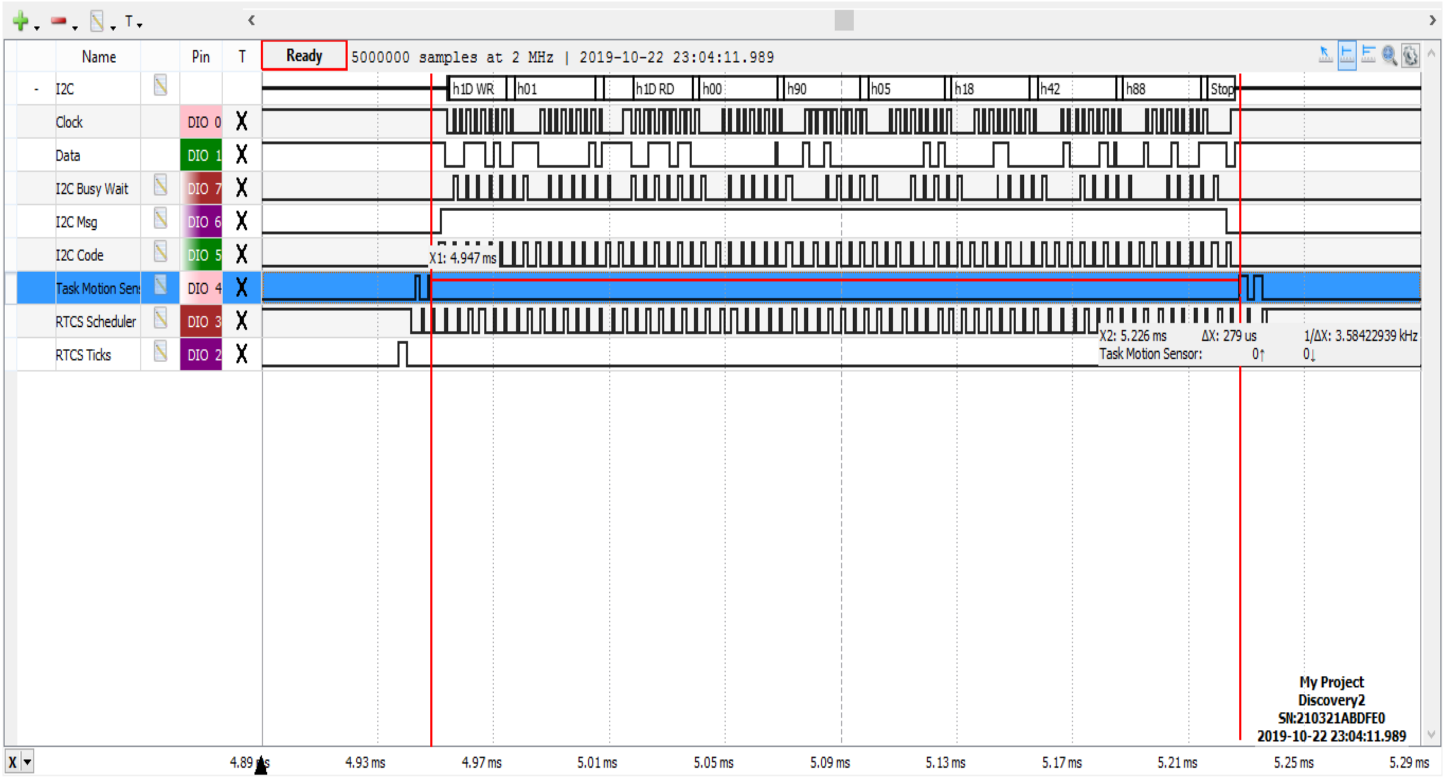


Ready 5000000 samples at 2 MHz | 2019-10-22 23:04:11.989

X: 4.95 ms

I2C Msg Pulse width: 271.5 us

My Project
Discovery2
SN:210321ABDFE0
2019-10-22 23:04:11.989



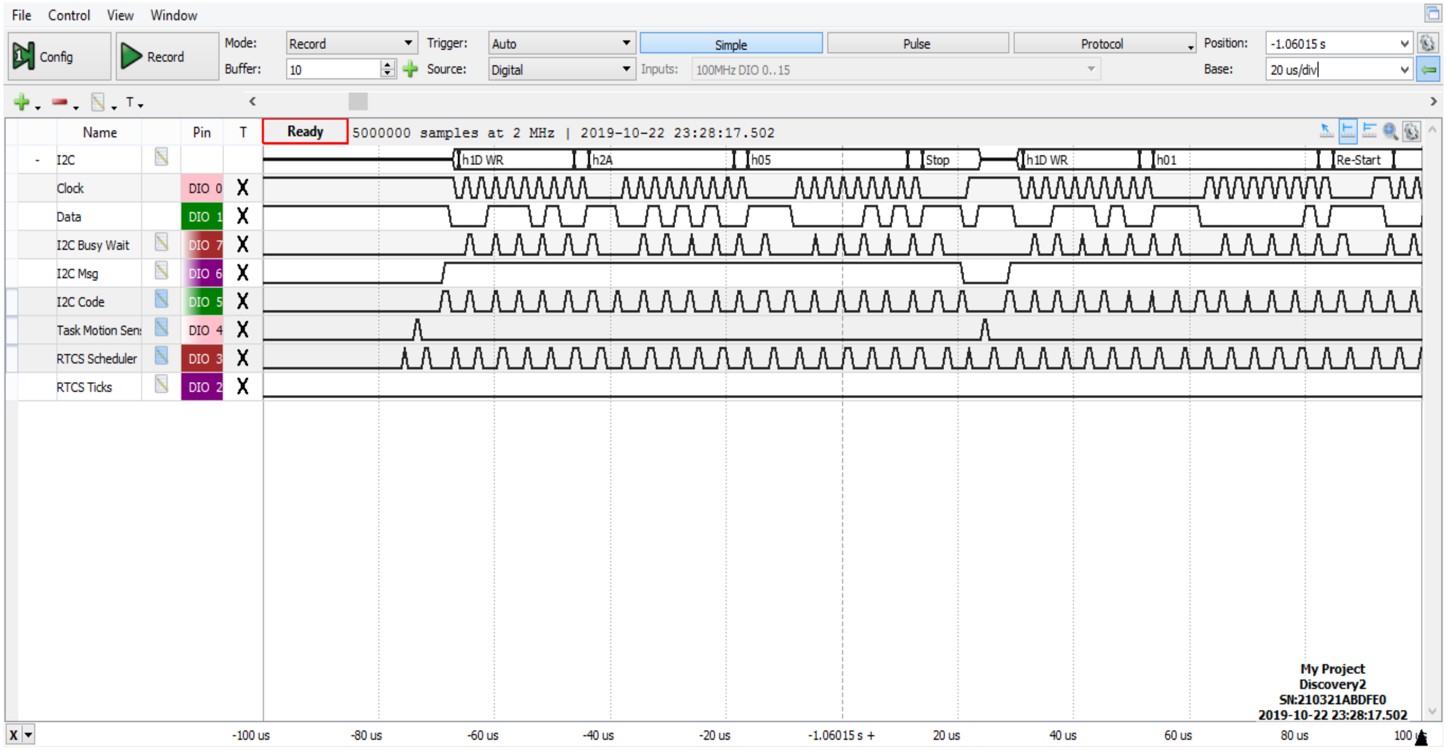
Ready 5000000 samples at 2 MHz | 2019-10-22 23:04:11.989

X1: 4.947 ms

X2: 5.226 ms ΔX: 279 us 1/ΔX: 3.58422939 kHz
Task Motion Sensor: 0↑ 0↓

My Project
Discovery2
SN:210321ABDFE0
2019-10-22 23:04:11.989

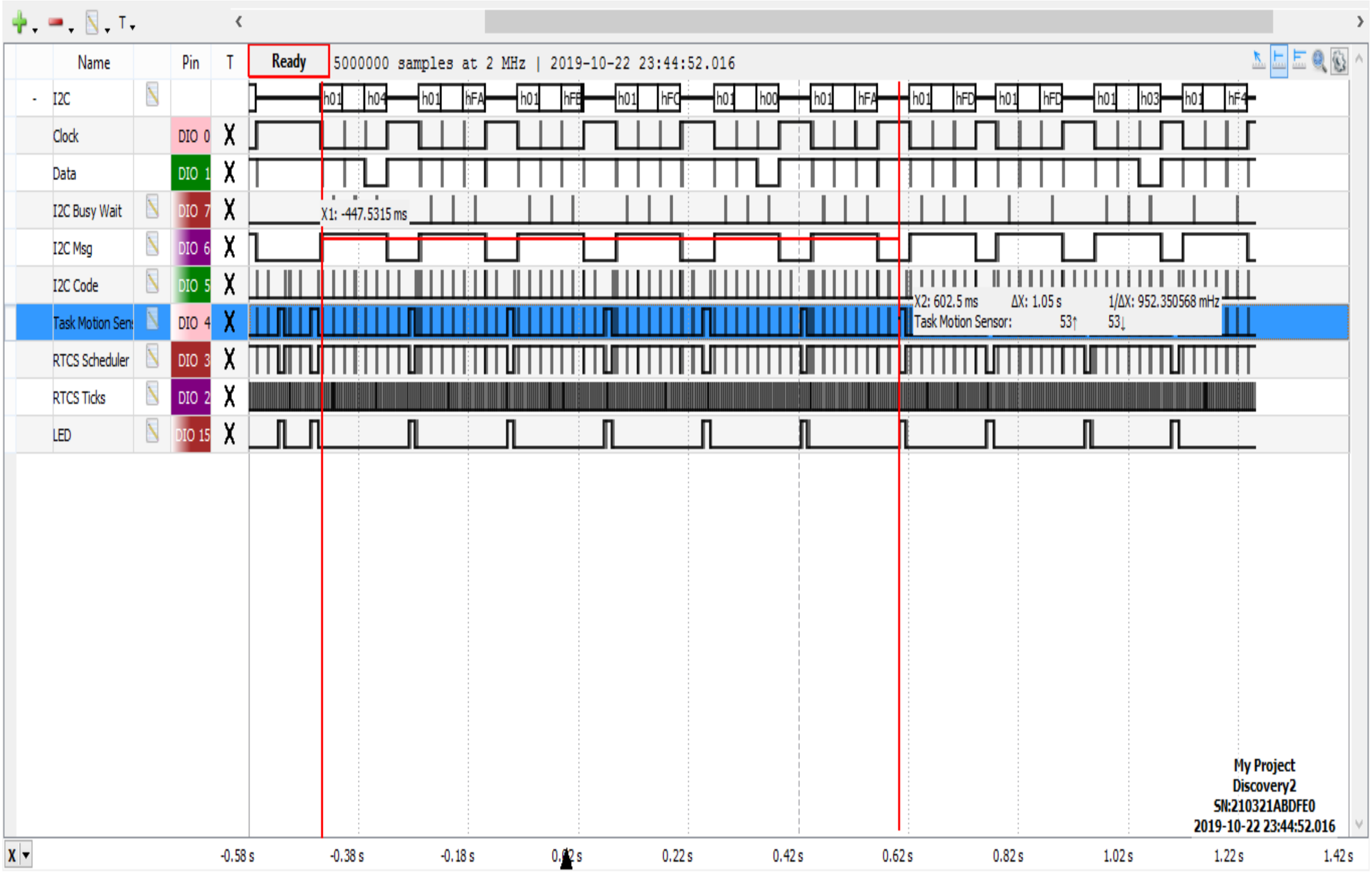
9. I2C Write – Event Triggering



Mode: Record
Buffer: 10
Trigger: Auto
Source: Digital
Inputs: 100MHz DIO 0..15
Position: -1.06015 s
Base: 20 us/div

Ready 5000000 samples at 2 MHz | 2019-10-22 23:28:17.502

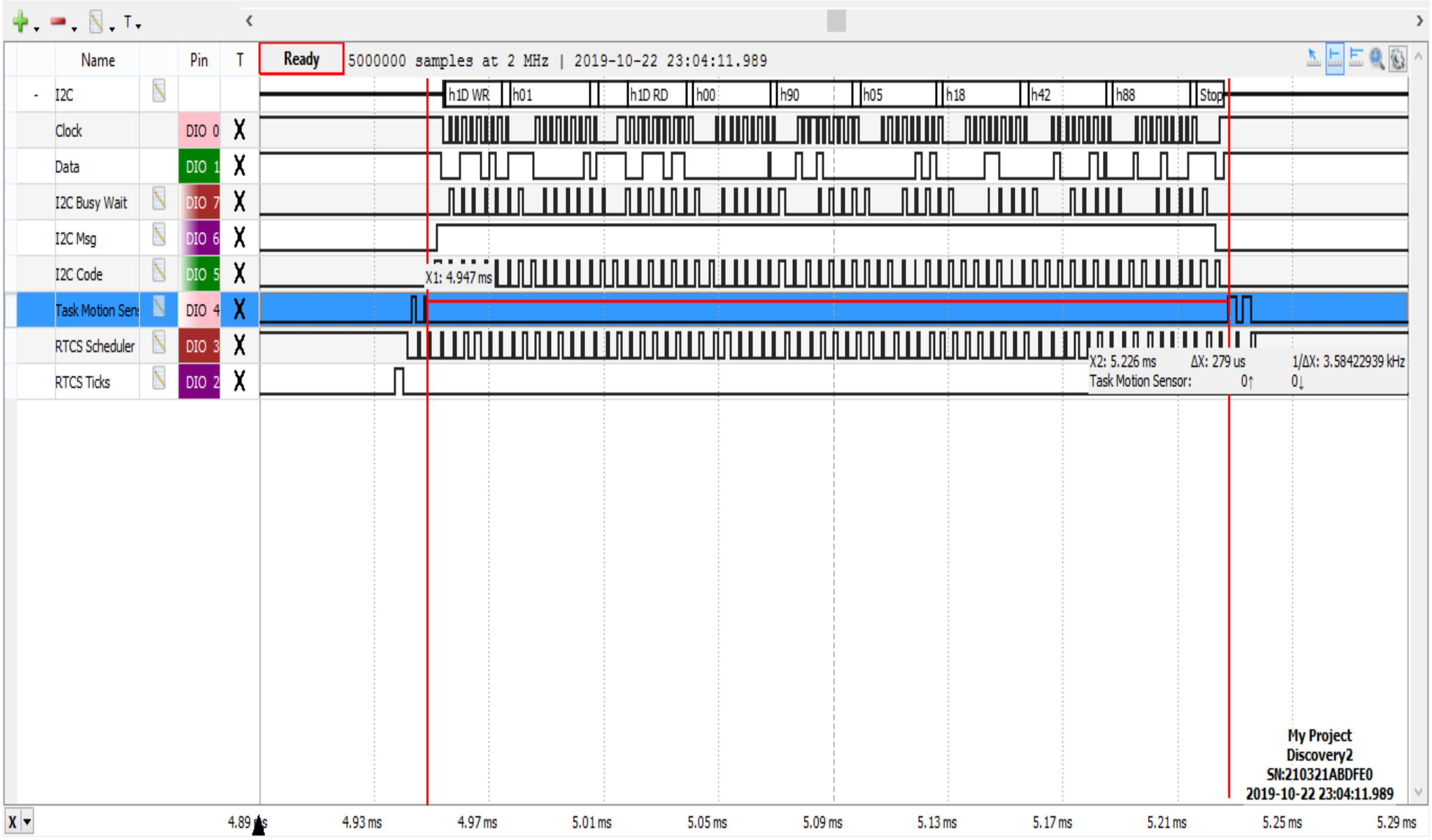
10.



Ready 5000000 samples at 2 MHz | 2019-10-22 23:44:52.016

X1: -447.5315 ms

X2: 602.5 ms ΔX: 1.05 s 1/ΔX: 952.350568 mHz
Task Motion Sensor: 53↑ 53↓



5000000 samples at 2 MHz | 2019-10-22 23:04:11.989

X1: 4.947 ms

X2: 5.226 ms ΔX: 279 us 1/ΔX: 3.58422939 kHz
Task Motion Sensor: 0↑ 0↓

My Project
Discovery2
SN:210321ABDFE0
2019-10-22 23:04:11.989

Improved code i.e. Event Triggered – Q8, timing approximately 279 us

Earlier code i.e. FSM – Q5, timing approximately 602.5 ms

Earlier code- Q5 has delay due to the period of each task. Each task is only run after period of each task, which increments release pending flags in RTCS Scheduler API. In event triggered, each task is released immediately after the task increasing the response.

It is evident from the following timing diagrams