Microprocessors and Microcontrollers Lab

DA - 4

Adarsh Shirawalmath - 22BKT0058

Aim:

i.) Generate two wave forms with duty cycles 40% and 70% using interrupts.

ii.) Generate a wave form with frequency 1 HZ and get data from P0 and sent to serial port.

Procedure:

i.) Start up the Keil µVision Software.

ii.) Create new µVision project at required directory.

iii.) Set the device as 8051 microcontroller (AT89C51).

iv.) Create new item at Source Group 1 in Target 1.

v.) Set the file type as ASM file.

vi.) Continue writing the code for the ALP.

vii.) Translate and build the file.

viii.) Start debug session, and run code line by line to get output

ix.) Check output at the memory location set, in memory 1.

Algorithm:

i.) Generate two wave forms with duty cycles 40% and 70% using interrupts.

1. **Initialization**:
    - Set the program's starting address to 0000H.
    - Jump to the main program (**MAIN**) using **LJMP MAIN**.
    - Set up interrupt handlers for INT0 and INT1 at addresses 0003H and 0013H, respectively.
2. **INT0 Interrupt Handler (WAVE_40)**:
    - Call the **WAVE_40** subroutine when INT0 is triggered (**ACALL WAVE_40**).
    - Return from interrupt (**RETI**).
3. **INT1 Interrupt Handler (WAVE_70)**:
    - Call the **WAVE_70** subroutine when INT1 is triggered (**ACALL WAVE_70**).
    - Return from interrupt (**RETI**).
4. **Main Program (MAIN)**:
    - Configure Timer 0 (TMOD) in mode 1 for 8-bit auto-reload.
    - Enable interrupts for INT0 and INT1 with specific priorities (INT1 - 70%, INT0 - 40%).
    - Enter an infinite loop (**SJMP $**) to continuously monitor interrupts.
5. **WAVE_40 Subroutine**:
    - Start a 40% duty cycle waveform on P2.3.
    - Call the **ON_40** subroutine to handle the ON time.
    - Turn off P2.3 using the **OFF_40** subroutine during the OFF time.
    - Repeat the waveform generation until INT0 is triggered.
    - Return from interrupt (**RETI**).
6. **WAVE_70 Subroutine**:
    - Start a 70% duty cycle waveform on P2.3.
    - Call the **ON_70** subroutine to handle the ON time.
    - Turn off P2.3 using the **OFF_70** subroutine during the OFF time.
    - Repeat the waveform generation until INT1 is triggered.
    - Return from interrupt (**RETI**).
7. **ON_40 Subroutine**:
    - Set Timer 0 for the ON time of the 40% duty cycle waveform.
    - Start Timer 0 and wait until it overflows (**JNB TF0, $**).

- Stop Timer 0 and clear its flag.
- Return from subroutine (**RET**).

8. **OFF_40 Subroutine**:
   - Set Timer 0 for the OFF time of the 40% duty cycle waveform.
   - Start Timer 0 and wait until it overflows (**JNB TF0, $**).
   - Stop Timer 0 and clear its flag.
   - Return from subroutine (**RET**).
9. **ON_70 Subroutine**:
   - Similar to **ON_40** but for the 70% duty cycle waveform.
10. **OFF_70 Subroutine**:
    - Similar to **OFF_40** but for the 70% duty cycle waveform.
11. **End of Program** (**END**).

ii.) Generate a wave form with frequency 1 HZ and get data from P0 and sent to serial port.:-

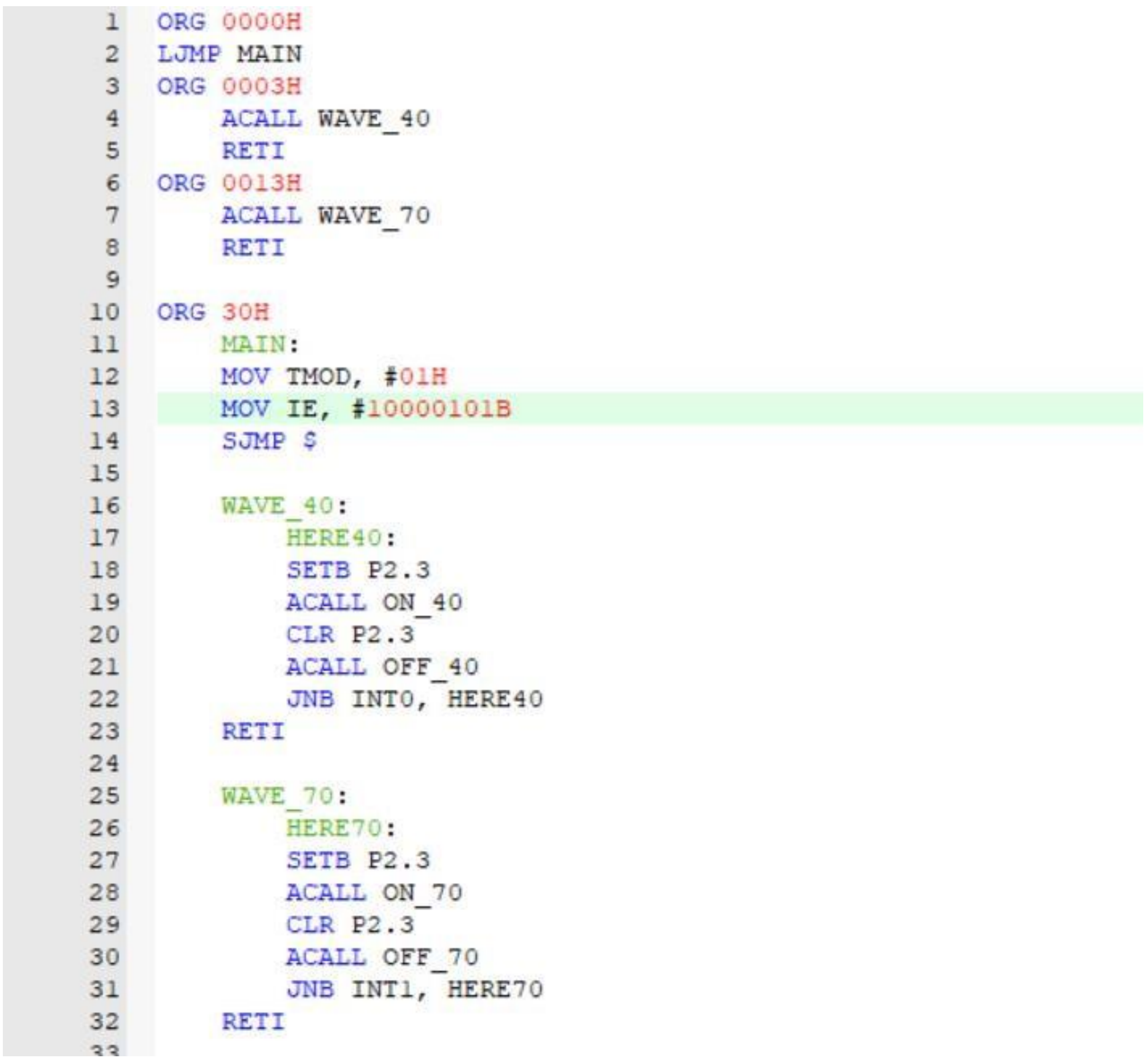
1. **Initialization**:
   - Set program counter to address 0010H and jump to **MAIN** using **LJMP MAIN**.
2. **INT0 Interrupt Handler (WAVE)**:
   - Call the **WAVE** subroutine when INT0 is triggered at address 0003H (**ACALL WAVE**).
   - Return from interrupt (**RETI**).
3. **MAIN**:
   - Configure Timer 0 (TMOD) in mode 2 (8-bit auto-reload) and Timer 1 for serial communication (**MOV TMOD, #20H**).
   - Set the serial control register (**MOV SCON, #50H**) and baud rate (**MOV TH1, #-3**).
   - Start Timer 1 (**SETB TR1**) and enable interrupts for INT0 with a specific priority (**MOV IE, #10000001B**).
   - Set P0 as an input port (**MOV P0, #11111111B**).
4. **Serial Data Transmission**:
   - Continuously send data from P0 to the serial port register (SBUF) in a loop.
   - Wait for transmission completion using the Transmit Interrupt (TI) flag.
5. **WAVE Subroutine**:
   - Toggle P2.3 at a specific frequency determined by the **DELAY** subroutine.
   - Repeat the waveform generation until INT0 is triggered.
   - Return from interrupt (**RETI**).
6. **DELAY Subroutine**:

- Set Timer 0 for a specific delay.
- Start Timer 0 and wait until it overflows (**JNB TF0, $**).
- Stop Timer 0 and clear its flag.
- Return from subroutine (**RET**).

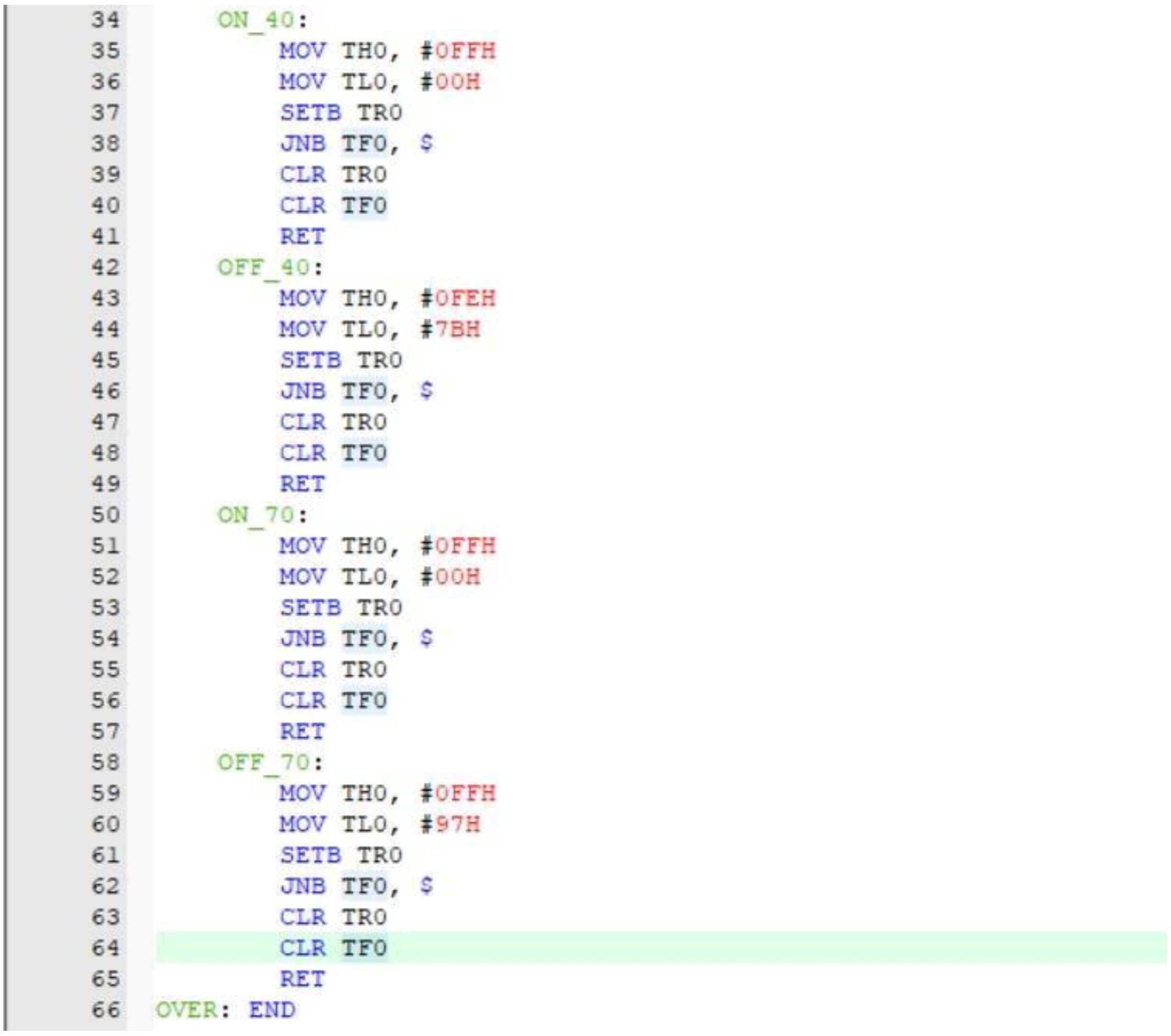
7. **End of Program** (**END**).

Code:

a.)

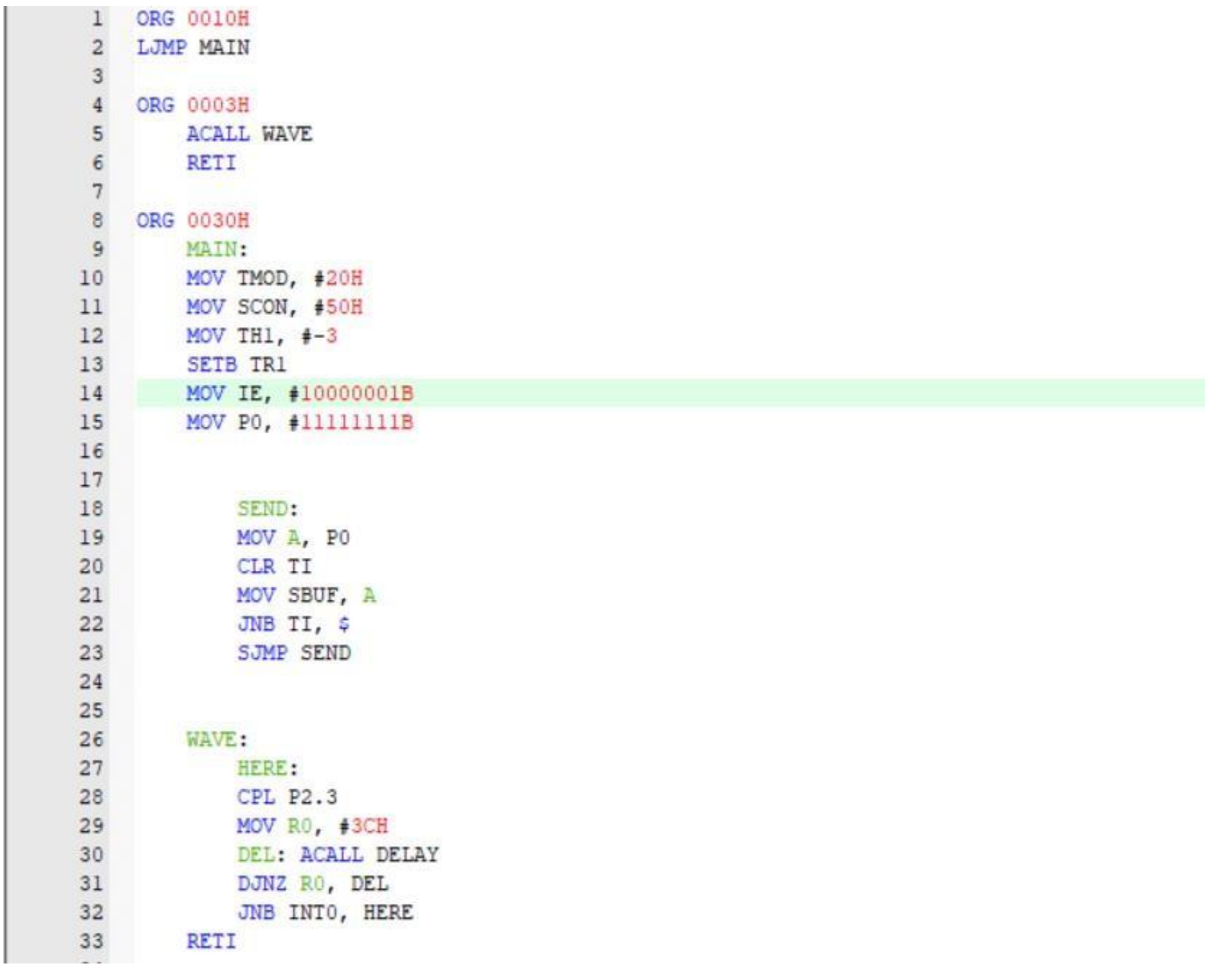


```
ORG 0000H
LJMP MAIN
ORG 0003H
    ACALL WAVE_40
    RETI
ORG 0013H
    ACALL WAVE_70
    RETI

ORG 30H
    MAIN:
    MOV TMOD, #01H
    MOV IE, #10000101B
    SJMP $

    WAVE_40:
        HERE40:
        SETB P2.3
        ACALL ON_40
        CLR P2.3
        ACALL OFF_40
        JNB INT0, HERE40
    RETI

    WAVE_70:
        HERE70:
        SETB P2.3
        ACALL ON_70
        CLR P2.3
        ACALL OFF_70
        JNB INT1, HERE70
    RETI
```

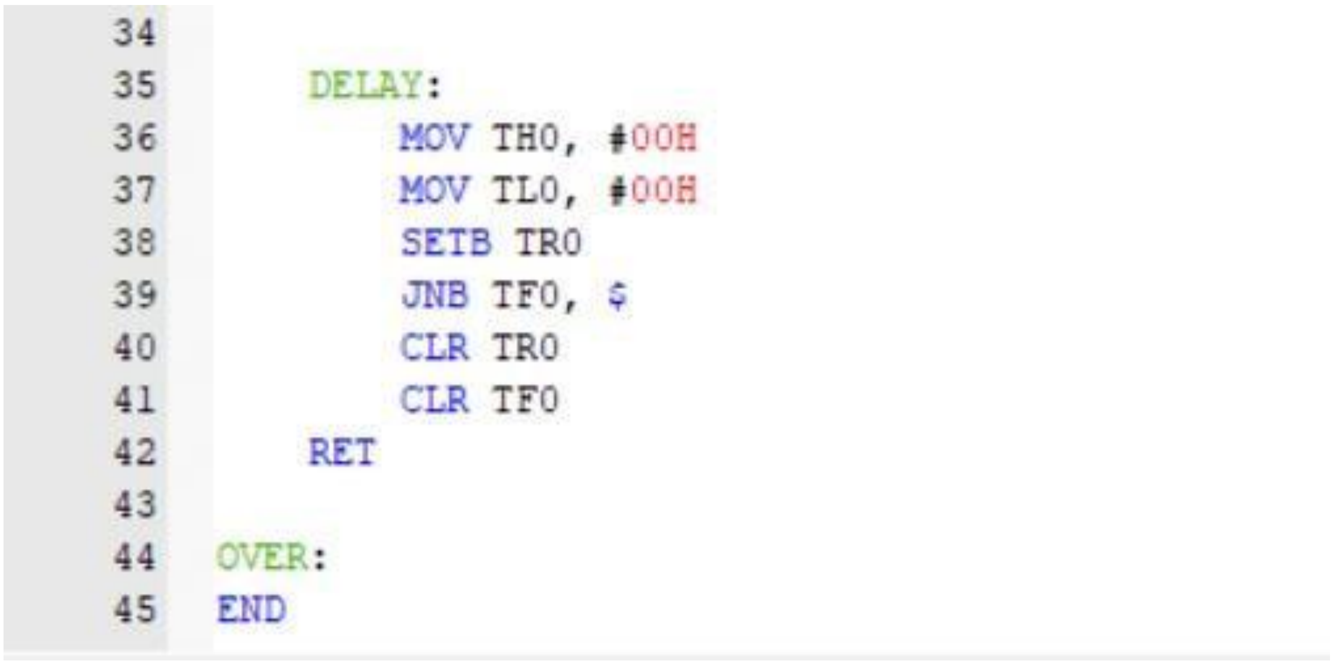


```
    ON_40:
        MOV TH0, #0FFH
        MOV TL0, #00H
        SETB TR0
        JNB TF0, $
        CLR TR0
        CLR TF0
        RET
    OFF_40:
        MOV TH0, #0FEH
        MOV TL0, #7BH
        SETB TR0
        JNB TF0, $
        CLR TR0
        CLR TF0
        RET
    ON_70:
        MOV TH0, #0FFH
        MOV TL0, #00H
        SETB TR0
        JNB TF0, $
        CLR TR0
        CLR TF0
        RET
    OFF_70:
        MOV TH0, #0FFH
        MOV TL0, #97H
        SETB TR0
        JNB TF0, $
        CLR TR0
        CLR TF0
        RET
OVER: END
```

b.)

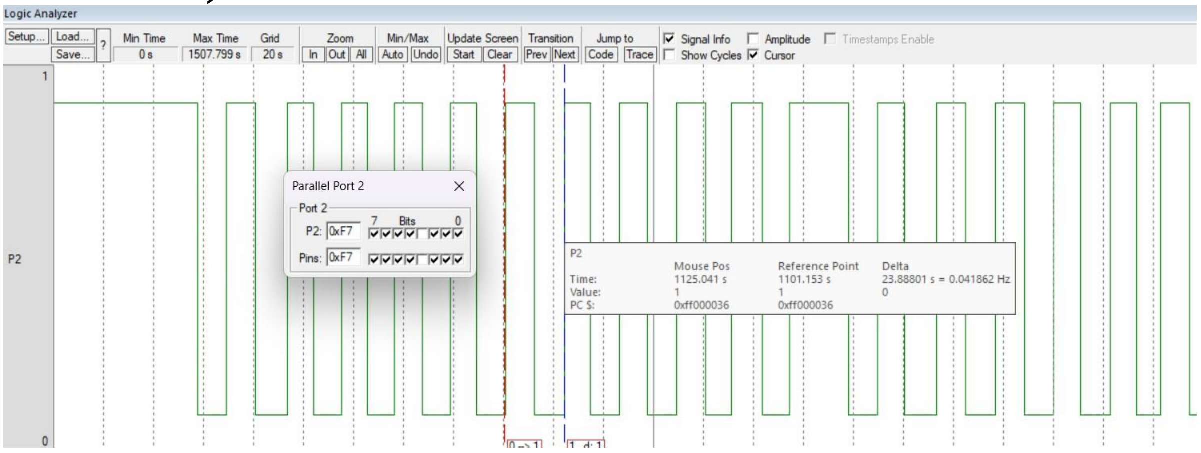


```
ORG 0010H
LJMP MAIN

ORG 0003H
    ACALL WAVE
    RETI

ORG 0030H
    MAIN:
    MOV TMOD, #20H
    MOV SCON, #50H
    MOV TH1, #-3
    SETB TR1
    MOV IE, #10000001B
    MOV P0, #11111111B


        SEND:
        MOV A, P0
        CLR TI
        MOV SBUF, A
        JNB TI, $
        SJMP SEND


    WAVE:
        HERE:
        CPL P2.3
        MOV R0, #3CH
        DEL: ACALL DELAY
        DJNZ R0, DEL
        JNB INT0, HERE
    RETI
```


```

    DELAY:
        MOV TH0, #00H
        MOV TL0, #00H
        SETB TR0
        JNB TF0, $
        CLR TR0
        CLR TF0
    RET

OVER:
END
```

Output:

a.)



Parallel Port 2

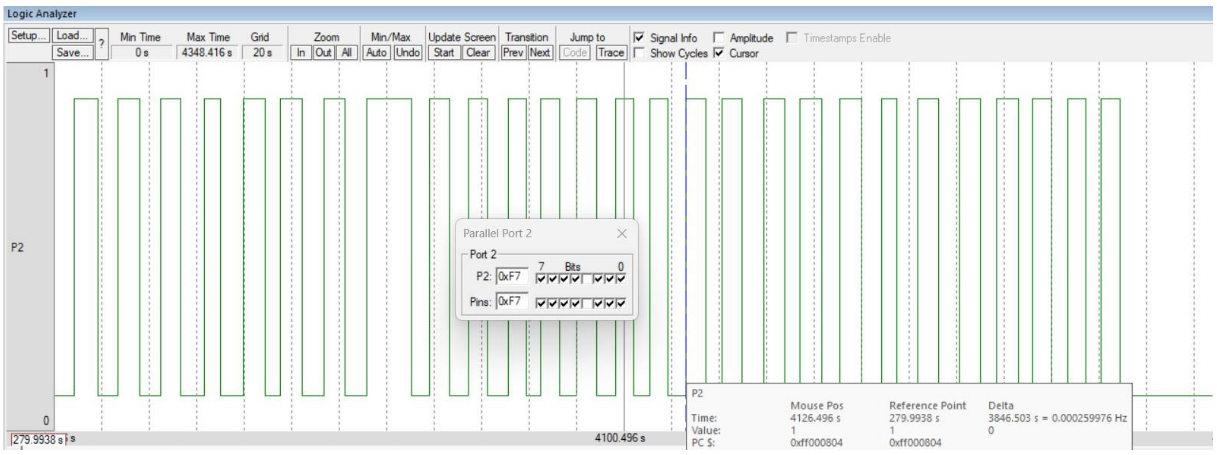
Port 2

P2: 0xF7 Bits 7 ☑☑☑☑☐☑☑☑ 0

Pins: 0xF7 ☑☑☑☑☐☑☑☑

```
P2
          Mouse Pos      Reference Point   Delta
Time:     1125.041 s     1101.153 s        23.88801 s = 0.041862 Hz
Value:    1              1                 0
PC $:     0xff000036     0xff000036
```

b.)



Min Time: 0 s
Max Time: 4348.416 s
Grid: 20 s

[✓] Signal Info
[ ] Amplitude
[ ] Timestamps Enable
[ ] Show Cycles
[✓] Cursor

Parallel Port 2

Port 2

		7		Bits					0
P2:	0xF7	✓	✓	✓	✓		✓	✓	✓
Pins:	0xF7	✓	✓	✓	✓		✓	✓	✓

```
P2
          Mouse Pos      Reference Point   Delta
Time:     4126.496 s     279.9938 s        3846.503 s = 0.000259976 Hz
Value:    1              1                 0
PC $:     0xff000804     0xff000804
```