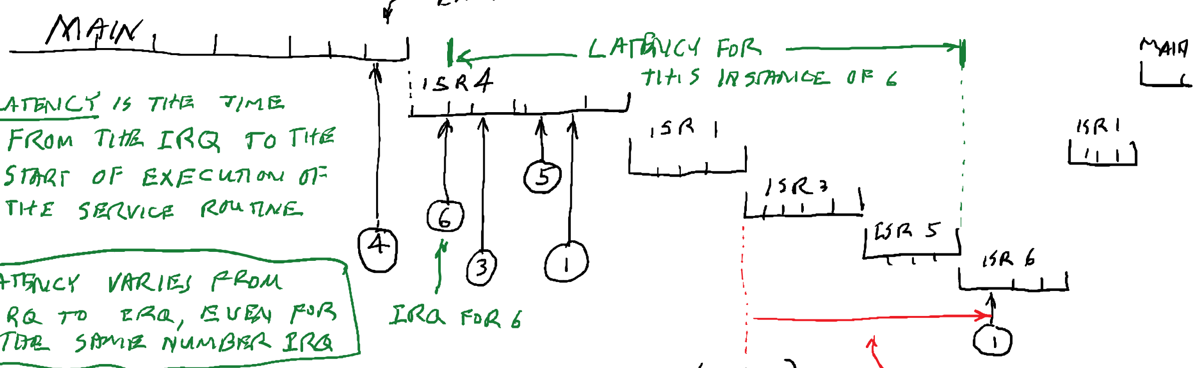
LATENCY, DENSITY, INTERRUPT INTERVALS

THE WAY INTERRUPTS MIGHT RUN

EACH TIC MARK IS A BOUNDARY BETWEEN INSTRUCTIONS (MACHINE CODE)

LATENCY IS THE TIME FROM THE IRQ TO THE START OF EXECUTION OF THE SERVICE ROUTINE

LATENCY VARIES FROM IRQ TO IRQ, EVEN FOR THE SAME NUMBER IRQ



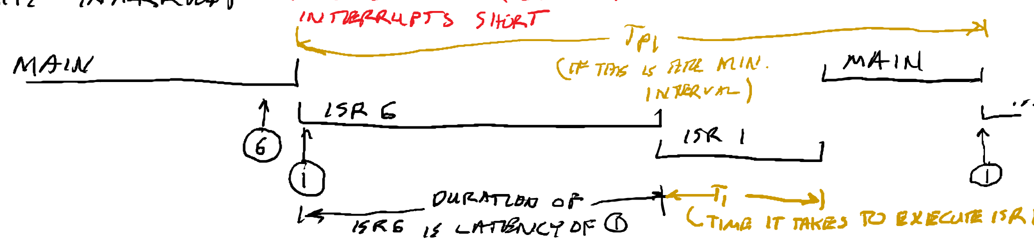
CIRCLED NUMBERS ARE INTERRUPT REQUEST (IRQ)

THEY HAVE A PRIORITY STRUCTURE, ① IS HIGHEST PRIORITY

→ TIME

1

- LOWER PRIORITY INTERRUPTS ARE LIKELY TO EXPERIENCE LONGER LATENCY → BUT NOT NECESSARILY WITH EVERY REQUEST
- ^ALL^ INTERRUPT SERVICE ROUTINES NEED TO BE SHORT BECAUSE THEIR DURATION CONTRIBUTES TO LATENCY
  - DO THE MINIMUM & DO IT EFFICIENTLY
  - PUSH AS MUCH WORK AS YOU CAN INTO MAIN & PROCEDURES THAT ARE INTERRUPTABLE
- MAX LATENCY FOR THE HIGHEST PRIORITY INTERRUPT IS AT LEAST THE DURATION OF THE LONGEST LOWER PRIORITY INTERRUPT
  - TO GIVE LOWER LATENCY KEEP LOW PRIORITY INTERRUPTS SHORT



2

INTERRUPT DENSITY RATIO

IS THE MAXIMUM FRACTION OR PERCENTAGE OF TIME THAT THE CPU COULD HAVE TO DEDICATE TO PROCESSING INTERRUPT SERVICE ROUTINES (ISR)

IRQ → HARDWARE SIGNAL
ISR → SOFTWARE THAT DOES THE WORK

FOR EACH INTERRUPT i ($i = 1$ IS HIGHEST PRIORITY)

DEFINE

DESIGN CONSTRAINTS NOT TYPICAL OR ACTUAL PERFORMANCE

T_i = THE LONGEST POSSIBLE TIME IT WILL TAKE TO EXECUTE THE ISR FOR INTERRUPT i

← HUMANS MUST ANALYZE THE CODE TO FIGURE THIS OUT

T_{P_i} = THE SHORTEST POSSIBLE TIME INTERVAL BETWEEN REQUESTS FOR ISR i (A.K.A "PERIOD")

EACH INTERRUPT HAS AN ASSOCIATED DENSITY RATIO $IDR_i = T_i / T_{P_i}$

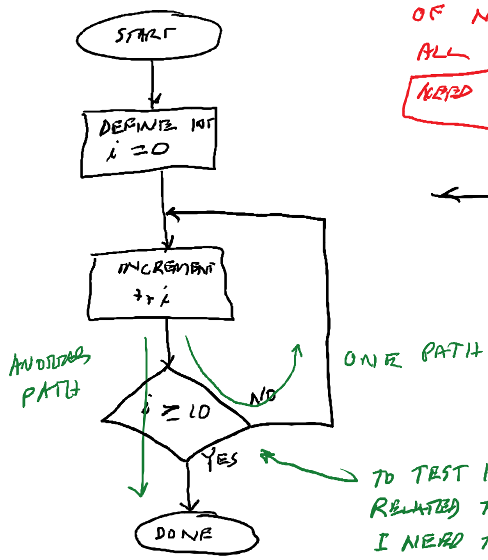
TOTAL IDR IS THE SUM $\sum_i IDR_i = \sum_i T_i / T_{P_i} < 1.00$ ← FOR RELIABILITY

3

SIDE ITEM — FLOW CHARTS

INTERRUPTS CREATE AN UNMANAGEABLE NUMBER OF NEW PATHWAYS — CAN'T POSSIBLY ITEMIZE ALL PATHS ON A FLOWCHART IF INTERRUPTS ARE INVOLVED

NEED AN ANALYTIC WAY TO MAKE SURE IT ALL WORKS!



←→

```
char i = 0;

for (i <= 10) {
    ++i;
}
```

↓

COMPILE TO MACHINE CODE

← TO TEST HARDWARE RELATED TO THIS PROCESSING I NEED TO TEST EACH PATH

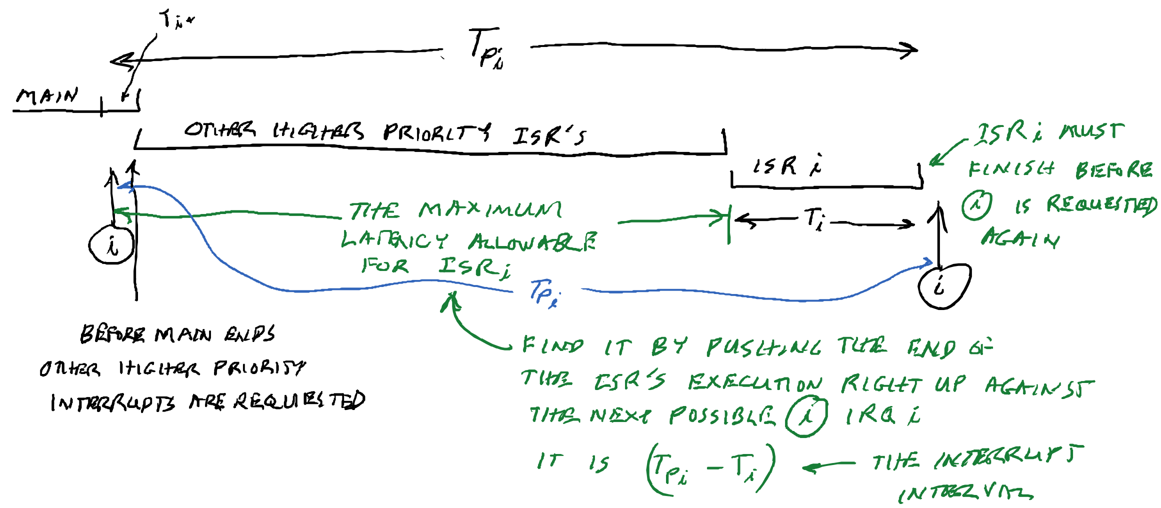
IF INTERRUPTS ARE ADDED, NEED TO TEST WITH & WITHOUT INTERRUPTS.

4

INTERRUPT INTERVALS — IS $(T_{P_i} - T_i)$

- NOT ONLY MUST THE CPU BE FAST ENOUGH TO HANDLE THE TOTAL INTR. DENSITY
- ISR_i MUST FINISH EXECUTION BEFORE IRQ_i CAN HAPPEN AGAIN

(ASSUMING THAT INTERRUPTS RUN WITH INTERRUPTS DISABLED)



MAKE SURE THAT THE INTERRUPT INTERVALS (ONE FOR EACH i) ARE NEVER EXCEEDED

DEFINE $N(i, x)$ = MAXIMUM NUMBER OF TIMES INTR. x (HIGHER PRIORITY) COULD BE REQUESTED DURING THE INTERRUPT INTERVAL FOR INTR. i

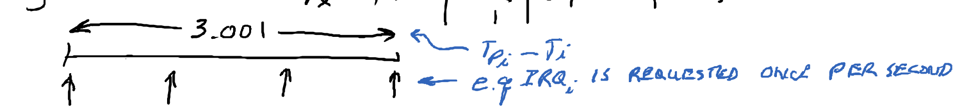
MAX TIME YOU CAN HOLD OFF $(T_{P_i} - T_i)$

$$N(i, x) = \left\lceil \frac{T_{P_i} - T_i}{T_{P_x}} \right\rceil$$

$\lceil \cdot \rceil$ ROUND UP TO INTEGER

$\lceil 3.56 \rceil = 4$

eg $T_{P_i} - T_i = 3.001$ $T_{P_x} = 1\,s$ $\left\lceil \frac{3.001}{1} \right\rceil = 4$ $\lceil 3.01 \rceil = 4$



4 REQUESTS WILL (JUST) FIT

CHECKING THE INTERRUPT INTERVAL FOR EACH (i)

IN THE LITERATURE → $T_{i+} + N(i,1)T_1 + N(i,2)T_2 + \dots + N(i,i)T_i < T_{P_i}$ ← FOR INTR (i)

eg $T_{3+} + N(3,1)T_1 + N(3,2)T_2 + N(3,3)T_3 < T_{P3}$ ← IF $i = 3$

SAME AS $T_{i+} + N(i,1)T_1 + N(i,2)T_2 + \dots < (T_{P_i} - T_i)$ $T_{P_i} - (1)T_i$

ALL THAT CAN CONTRIBUTE TO LATENCY

MUST NOT EXCEED THIS INTR INTERVAL

FIRST ANALYZE CODE & DEFINE EACH T_{P_i} AND T_i

T_{i+} IS COMPUTED FOR EACH INTR i

IS LONGEST OF

- THE LONGEST MACHINE INSTRUCTION IN THE CODE
- OR THE LONGEST LOWER PRIORITY INTERRUPT'S T_i
- OR THE LONGEST CRITICAL REGION

$N(i,i) = 1$

$$N(i,i) = \left\lceil \frac{T_{P_i} - T_i}{T_{P_i}} \right\rceil = 1$$

CAN PROVE $N(i,i) = 1$

7

Interrupt example, Three interrupts (Interrupt 1 is highest priority)

There are no critical regions,
the longest instruction takes 1 ms to execute
Interrupts run with interrupts disabled
For interrupt #1, T_{P1} = 60 ms, T_1 = 1.0 ms $T_{1+} = 2.5$ ms
For interrupt #2, T_{P2} = 20 ms, T_2 = 2.5 ms $T_{2+} = 1$ ms
For interrupt #3, T_{P3} = 4 ms, T_3 = 1.0 ms $T_{3+} = 1$ ms

STEP (1) CHECK INTERRUPT DENSITY RATIO

$$\sum_i \frac{T_i}{T_{P_i}} = \frac{1\text{ ms}}{60\text{ ms}} + \frac{2.5}{20} + \frac{1}{4} = 0.392 < 1.0 \quad \text{(OK)}$$

∴ MUST CHECK INTERRUPT INTERVALS

STEP (2) EVALUATE T_{i+} VALUES (START W/ LOW)

STEP (3) APPLY INTERVAL FORMULA $T_{i+} + N(i,1)T_1 + N(i,2)T_2 + \dots N(i,i)T_i < T_{P_i}$

FOR INTR. #1 $T_{1+} + N(1,1)T_1 < P_{P_1}$ → $2.5\text{ ms} + (1)\,1\text{ ms} < 60\text{ ms}$ → $3.5 < 60$ (OK)

FOR INTR #2 $T_{2+} + N(2,1)T_1 + N(2,2)T_2 < T_{P2}$ → $1\text{ ms} + (1)\,1\text{ ms} + (1)\,2.5\text{ ms} < 20\text{ ms}$ (OK)

$$N(i,x) = \left\lceil \frac{T_{P_i} - T_i}{T_{P_x}} \right\rceil \quad N(2,1) = \left\lceil \frac{T_{P2} - T_2}{T_{P1}} \right\rceil = \left\lceil \frac{20\text{ ms} - 2.5\text{ ms}}{60\text{ ms}} \right\rceil = 1$$

FOR INTR #3 $T_{3+} + N(3,1)T_1 + N(3,2)T_2 + N(3,3)T_3 < T_{P3}$

$1\text{ ms} + (1)\,1\text{ ms} + (1)\,2.5\text{ ms} + (1)\,1\text{ ms} < 4\text{ ms}$

$5.5 \not< 4$

FAILURE!

$$N(3,1) = \left\lceil \frac{T_{P3} - T_3}{T_{P1}} \right\rceil = \left\lceil \frac{4-1}{60} \right\rceil = 1$$

$$N(3,2) = \left\lceil \frac{T_{P3} - T_3}{T_{P2}} \right\rceil = \left\lceil \frac{4-1}{20} \right\rceil = 1$$

8