

CSE 467S Homework #1

1. SHARC

- (15 pts) List and briefly explain SHARC's three special features designed to speed up the FIR filter (excluding function calls).
- (10 pts) Please show the states of (i) the PC stack, (ii) the loop address stack, and (iii) the loop counter stack in the end of every iteration of the following nested loop (including both outer and inner loops)?

```
S1:  LCNTR=2, DO LP2 UNTIL LCE;
S2:  LCNTR=3, DO LP1 UNTIL LCE;
      R1=DM(I0,M0), R2=PM(I8,M8);
LP1:  R8=R1*R2;
      R12=R12+R8;
LP2:  R11=R11+R12;
```

- Dynamic Power Management. Suppose a workload trace on a SA-1100 processor when DPM is *not* applied is as follows:

15, 40, 300, 300, 10, 1000, 50, 110, 1200 (ms)

Non-underlined numbers refer to the duration of an inactive period, and underlined numbers refer to the duration of an active period. We assume the processor starts in the RUN state. Refer to class slides and the DPM paper for SA-1100's Power State Machine and power-related parameters. For simplicity, we ignore the transition time from RUN to SLEEP, i.e., assume it takes 0 sec. Only the SLEEP state is used for power saving. The IDLE state is not used.

What are the (i) **total run time** (including the time the processor spends in RUN, SLEEP, and Transition), and (ii) **total energy consumption** if each of the following DPM policies are used?

Note: You **must** show how you get your final answer by writing down the important intermediate steps in your solution.

- (10 pts) Fixed time-out with a timeout threshold of 40 ms.
- (10 pts) Fixed time-out with a timeout threshold of 350 ms.
- (15 pts) Threshold-based predictive shutdown with a threshold of 60 ms?
- (10 pts) If the workload is performance critical (but less energy constrained), which policy should we use? What if the workload is energy constrained (but less performance critical)? The analysis should be based on the above quantitative analysis.

- Function call: In our example of ARM7 function calls (slide 13 on program optimization) each function call adds 3 instructions to the caller and 3 instructions to the callee. Will the code size of the following programs increase if we apply function inlining to every function call in the following programs?

Note: You must show how you get your answer through quantitative analysis. You will receive no point if you do not provide the analysis.

- (15 pts) The program has one function, trick1(int x). The body of the function has 4 instructions. The function is called 6 times.
- (15 pts) The program has one function, trick2(int y). The body of the function has 25 instructions. The function is called three times.