UNIVERSITY OF TEXAS AT AUSTIN
Dept. of Electrical and Computer Engineering

EE382C-9 Embedded Software Systems
Problem Set #3: More Models of Computation

Date assigned:	April 20, 2004
Date due:	April 26, 2004

Late homework will not be accepted.

Reading: *Languages for Embedded Systems*, ch. 12 and 17

You may use any computer program to help you solve these problems, check answers, etc.

You will need to run the full version of Ptolemy for this homework set.

In order to fix the problems in running the Xgraph blocks, set your version of the Java Developer's Kit (JDK) to 1.1 by setting the following environment variable in your login scripts and re-execute your login scripts:

```
setenv JAVA /usr/local/packages/java1.1
```

I currently have my ~bevans/.mycshrc file set for version 1.1 of the Java Developer's Kit, and you are welcome to copy the file.

Problem 3.1 Code Generation Domains

The Motorola 56000 is a modified Harvard architecture with three memory banks: one for programs (p) and two for data (x and y). Open the CG56 (Code Generation for the 56000) demonstration under the Simulator ... DTMFCodec demonstration. DTMF stands for dual-tone multi-frequency, which are tones generated by touchtone telephones. Edit the target and set the target parameter 'show run time?' to `YES`. Program and data memory usage should be reported by default. Run the DTMF detector demonstration.

a. How much p, x, and y memory does the DTMF decoder require?

b. What is the execution time in instruction cycles? Note that the results of the simulation won't be plotted unless you have your JAVA environment variable set to JDK 1.0.2. Please see the hints at the beginning of the assignment.

c. What is the execution time in seconds for a standard 56000 processor? Note that the Motorola 56000 uses two clock cycles to execute one instruction cycle. For standard clock speeds, you can consult the Berkeley Design Technology Inc. pocket guide to DSP processors http://www.bdti.com/pocket/pocket.htm.

d. On how many telephone channels could this implementation on a 56000 perform DTMF detection in real-time at the same time? The DTMF detection demonstration runs for

4 iterations. One way to find out how many input samples are processed in each iteration is to look at the schedule and find the outer repeat block and determine the number of tokens produced by that source block per firing. In this case, the source block produces one token per invocation. The standard sampling rate for a telephone line is 8000 samples/s.

For more information on DTMF tones, see slide 3 of lecture 3 for the real-time DSP lab course at

http://www.ece.utexas.edu/˜bevans/courses/realtime/lectures/03_Signals_Systems

Problem 3.2 Comparing Domains
In Ptolemy Classic, run the Basic ... Butterfly SDF demonstrations in the SDF and CGC domains and compare their run times. Comment on the differences.

Problem 3.3 Retiming
In the Ptolemy code generation (CG) domain demonstrations, run the retiming demonstration called `pipeline`. You can ignore the warnings about "use of variable delays". The initial Gantt chart shows that processors 2 and 3 are idle 11 out of every 12 cycles. You can click on any processor time slot and the blocks that are active during that time slot are the ones that execute during that time slot. Now, change the universe parameter `retime` to YES and run the demonstration again.

- What are the production and consumption on the arcs of the graph? Compute the repetitions vector for the graph.
- Without retiming, what is the utilization of the three processors? What are the memory requirements for the buffers on the arcs?
- With retiming, what is the utilization of the three processors? What are the memory requirements for the buffers on the arcs? Find the minimum buffer memory requirements for the same utilization by directly setting the amount of delay on each feedforward arc.
- For your minimum buffer solution, draw an acyclic precedence graph.
- Does retiming affect the computation of the repetitions vector?

Problem 3.4 Extend Block Library in Ptolemy Classic
Create and dynamically link a star in Ptolemy called SDFMyFIR.pl that will implement the finite impulse filter you wrote in C++ for the first homework assignment. When the FIR filter filter executes, the star should consume and produce one token. Create a simple schematic that connects an IIDGausian source star to the input of the FIR filter. Also, connect a simple plotter to the output of the filter. Turn in a print out of the result. Submit SDFMyFIR.pl.
I have made a template for an SDFMyFIR.pl star available at

http://www.ece.utexas.edu/~bevans/courses/ee382c/homework/SDFMyFIR.pl

To define prototypes for functions in standard C++ libraries, you can add the following to the .pl file.

```
ccinclude { <stdio.h>, <stdlib.h> }
```

To compile your star, you will need to do two steps:

(a) Run the preprocessor to convert the .pl file into .cc and .h files:

```
ptlang SDFMyFIR.pl
```

(b) Run the C++ compiler that was used to build Ptolemy:

```
g++ -I$PTOLEMY/src/kernel -I$PTOLEMY/src/domains/sdf/kernel -c SDFMyFIR.cc
```

You should be using the following version of the C++ compiler:

```
tick.ece 368# which g++
/usr/local/packages/ptolemy0.7.1/bin.sol2.5/g++
```

Once you've loaded a new star, Ptolemy

- will not detect later changes you've make to the star so it won't automatically relink the new version of the star
- will not let you manually relink a new version of the star using either 'make-star' ('*' shortcut) or 'Load' ('L' shortcut) commands.

So, you'll have to quit and restart Ptolemy to load a new version of a star. Some would call this a feature. :-)