PRINCE OF SONGKLA UNIVERSITY
FACULTY OF ENIGNEERING

Final Examination : Semester I — Academic Year : 2006
Date : 10 October 2006 — Time: 9:00 – 12:00
Subject : 240- 534 ADVANCED COMPUTER ENGINEERING DESIGN AND SYSTEMS — Room : R200

Text book and documents are allowed.

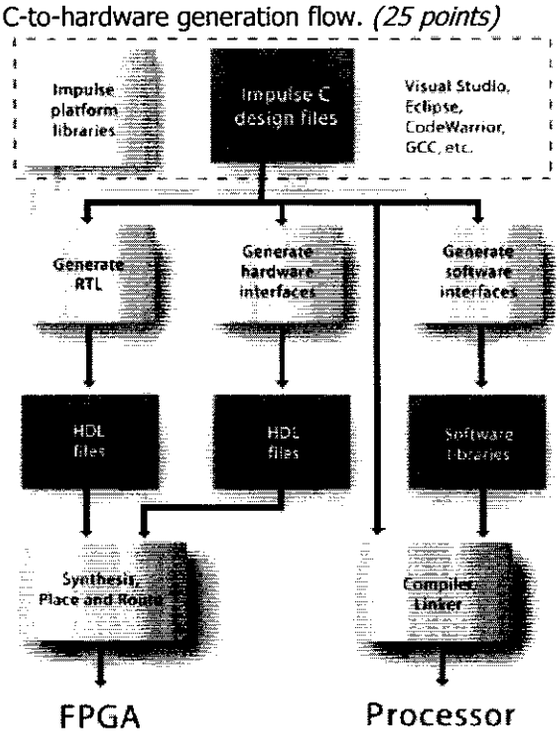
1. Please explain the C-to-hardware generation flow. *(25 points)*
2. What's Pragma CO Pipeline & Pragma Co Unroll? (10 points)
3. Design hardware accelerator for DES algorithm using Embedded FPGA (40 points)
    a. Diagram of hardware accelerator DES Architecture (10 points)
    b. How to build software for systems testing (10 points)
    c. Hardware process implemented by Impulse C (20 points)

PRINCE OF SONGKLA UNIVERSITY
FACULTY OF ENIGNEERING

Final Examination : Semester I — Academic Year : 2006
Date : 3 August 2006 — Time:
Subject : 240- 534 ADVANCED COMPUTER ENGINEERING DESIGN AND SYSTEMS — Room :

---

1. Please explain the C-to-hardware generation flow. *(25 points)*



In this flow, design entry and initial desktop simulation and debugging are performed using common C development tools such as Visual Studio or gcc and gdb. The Impulse libraries provide the needed functions (as described in the preceding chapters) for software emulation of parallel processing.

Once an application has been developed and simulated in the context of standard C programming tools, it can then be targeted to a specific platform and compiled into hardware and software binary files. For software functions (those that will be compiled to an embedded processor, for example) the process is relatively straightforward and uses standard cross-compiler tools in conjunction with platform-specific runtime libraries provided with the Impulse compiler.

2. What's Pragma CO Pipeline & Pragma Co Unroll?, give explain.

Pragma CO PIPELINE

Pipelining of instructions is not automatic and requires an explicit declaration in your C source code as follows:

```
#pragma CO PIPELINE
```

This declaration must be included within the body of the loop and prior to any statements that are to be pipelined. For example:

```
for (i=0; i<10; i++) {
#pragma CO PIPELINE
  sum += i << 1;
}
```

Note

The PIPELINE pragma must appear at the top of the loop to be pipelined, before any other statements within the loop, and the loop may not contain any nested loops.

Pragma CO UNROLL

Loop unrolling may be enabled with the use of the UNROLL pragma, which appears as follows:

```
for (tap = 0; tap < TAPS; tap++) {
#pragma CO UNROLL
    accum += firbuffer[tap] * coef[tap];
}
```

Unrolling a loop will result in that code within the loop being duplicated in hardware as many times as needed to implement the operation being described. It is therefore important to consider the size of the resulting hardware and unroll loops that have a relatively small number of iterations. The iterations of the loop must also not include interdependent calculations and/or assignments that would prevent the loop from being implemented as a parallel (unrolled) structure in hardware.

Note that the UNROLL pragma must appear at the top of the loop, before any other statements in the loop, and the loop must be a `for` loop with a loop variable of type `int` and constant bounds.

3. Depends on student's design.