Architecture, Design and Implementation of Embedded Systems for Real-Time Applications

CpE-450 Spring 05

Class 2(3)

Bruce McNair

bmcnair@stevens.edu

Syllabus - by week

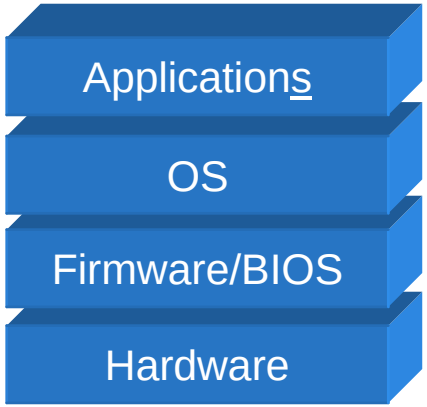
1. Introduction
   - Definition of embedded system
   - Constraints on embedded vs. standalone systems
   - Concept of real-time design
   - Time scales for real-time systems
   - Applications
2. Hardware/software functional partitioning
   - Relevant hardware technologies:
     - Discrete logic
     - CPLDs, FPGAs, ASICs
   - Software environments
     - HLL vs. assembly coding
     - DSP vs. general purpose vs. RISC
3. Development environments, course project definition
4. System architectures
5. Pipelining, interrupt service routines

Definition of Terms

- Embedded System – a computing system that is *embedded* within a larger system. The embedded system may or may not be visible to the end user, but it's function is not the intended end application of the user-perceived system.

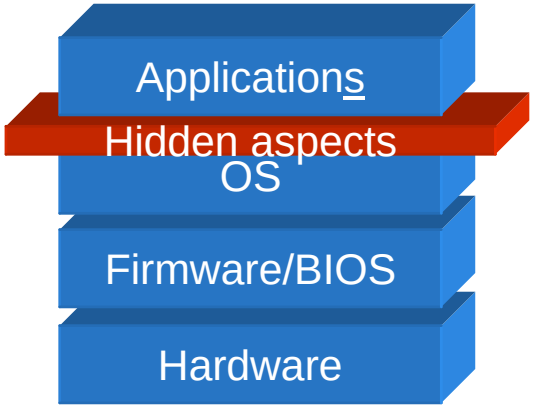
- Real-time Application – an application that has specific time-bound performance requirements

System Abstraction



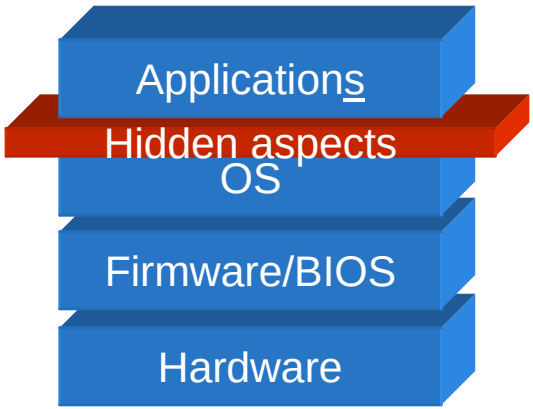
Standalone System

System Abstraction

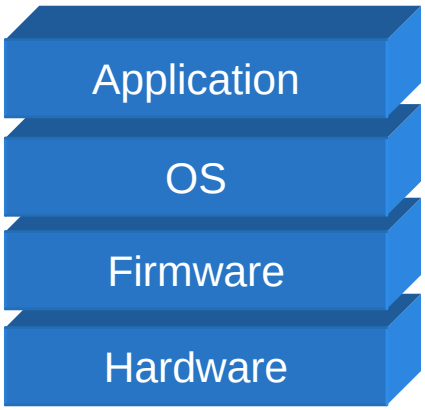


Standalone System

System Abstraction

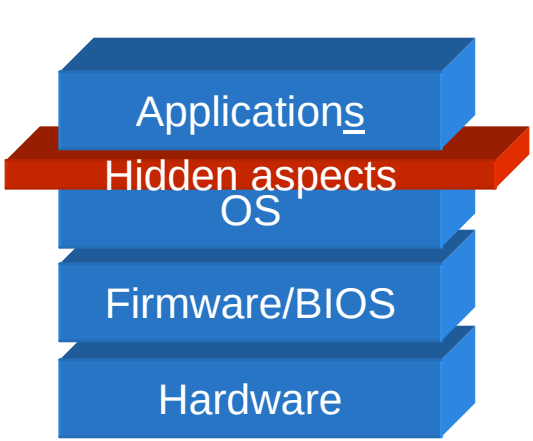


Standalone System

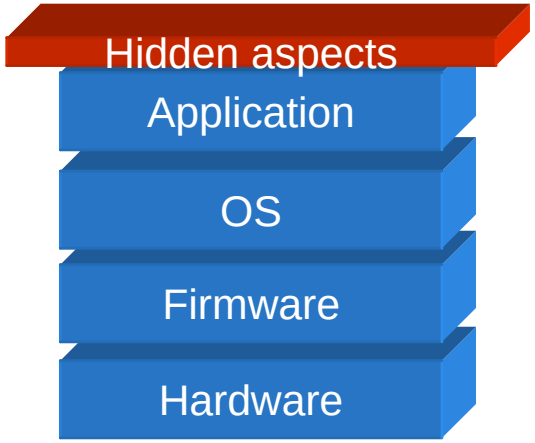


Complex Embedded System

System Abstraction

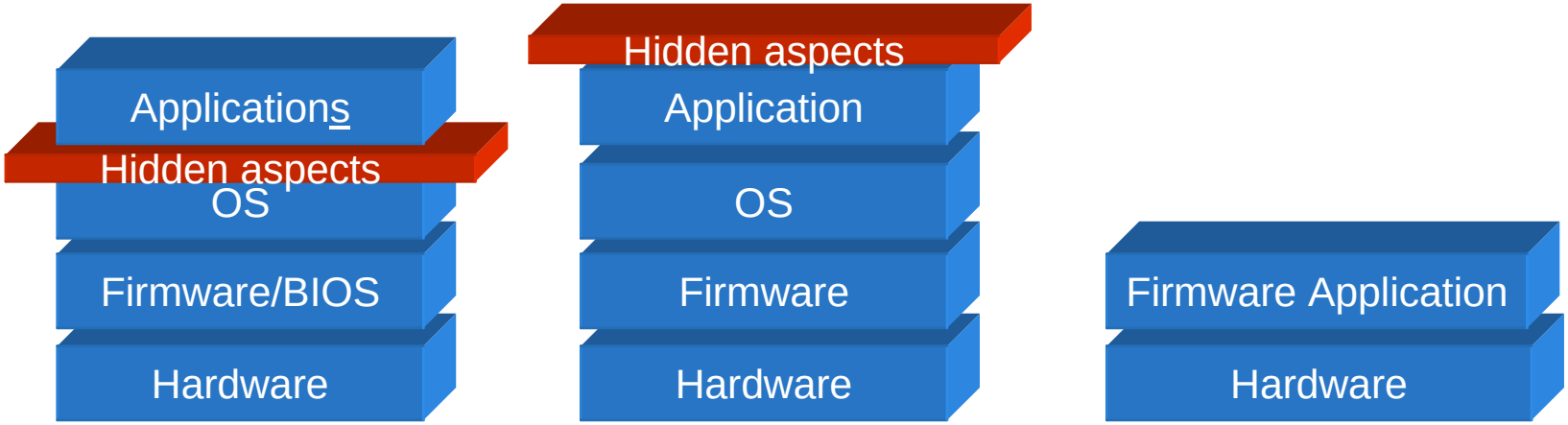


Standalone System



Complex Embedded System

System Abstraction

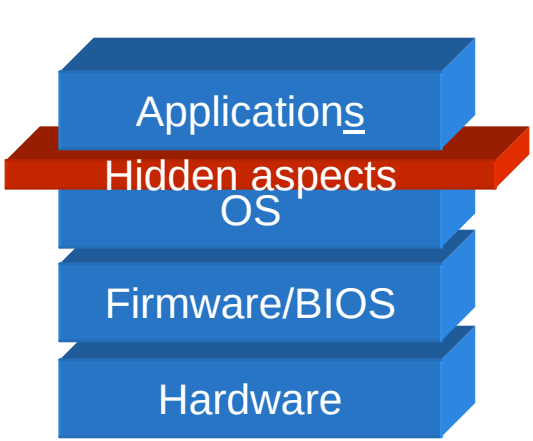


Standalone System

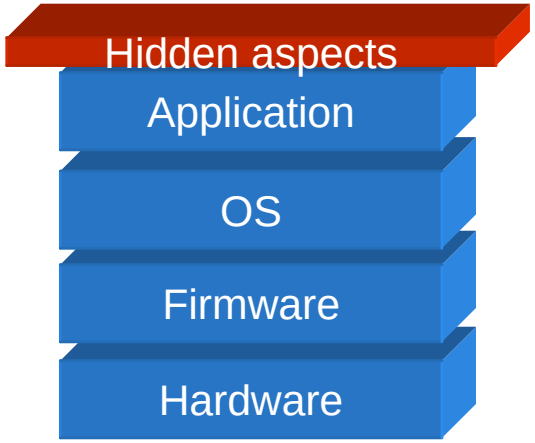
Complex Embedded System

Basic Embedded System

System Abstraction



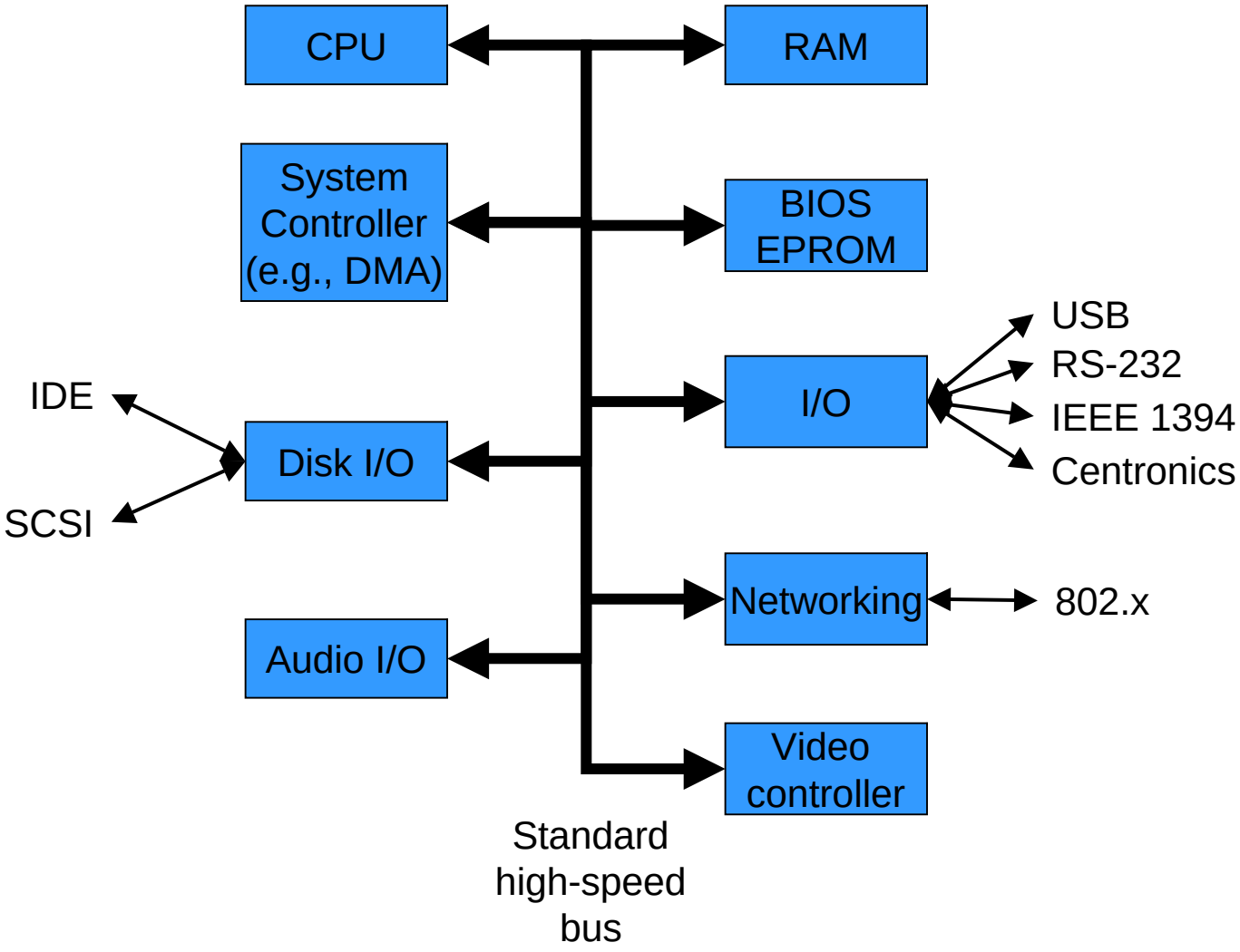
Standalone System



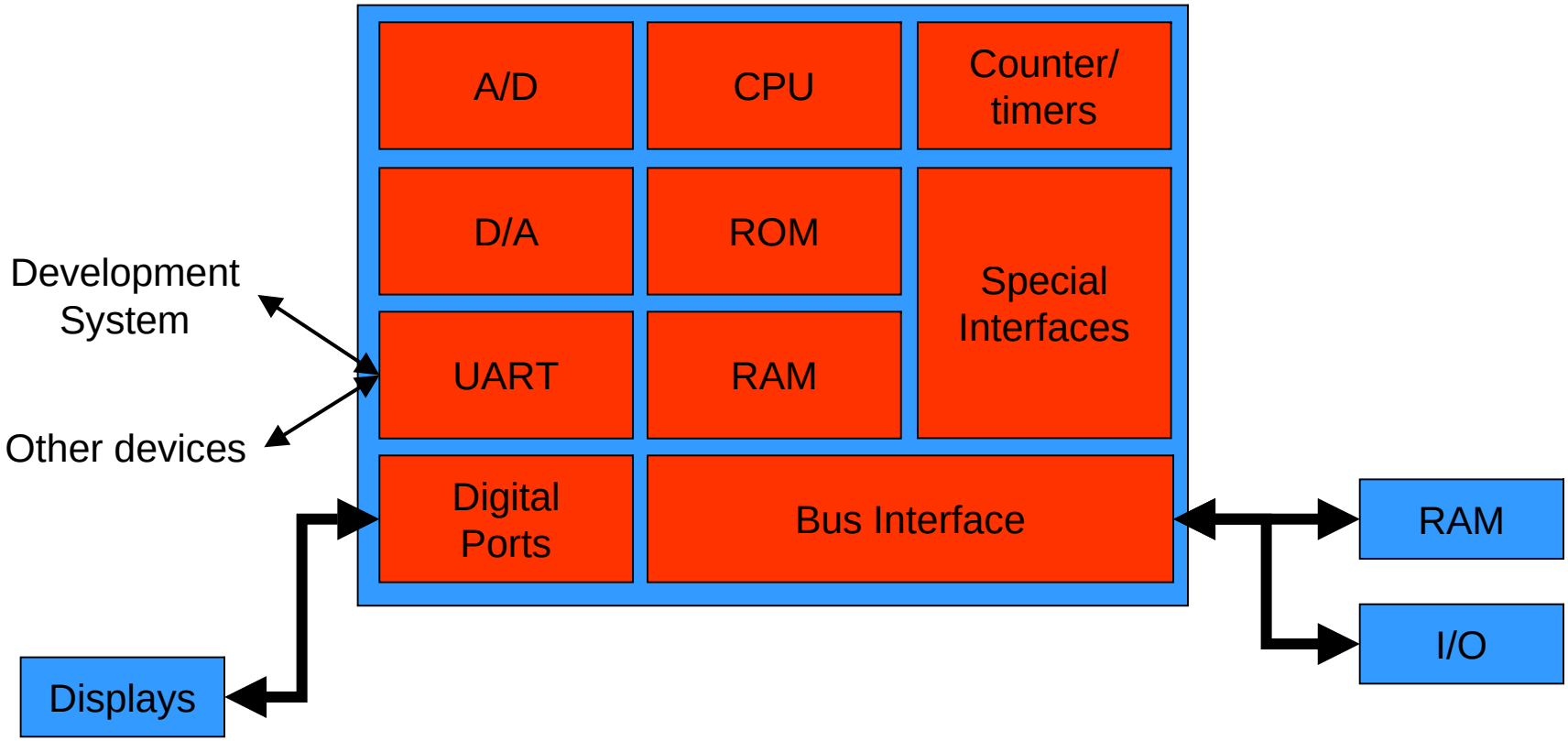
Complex Embedded System

Basic Embedded System

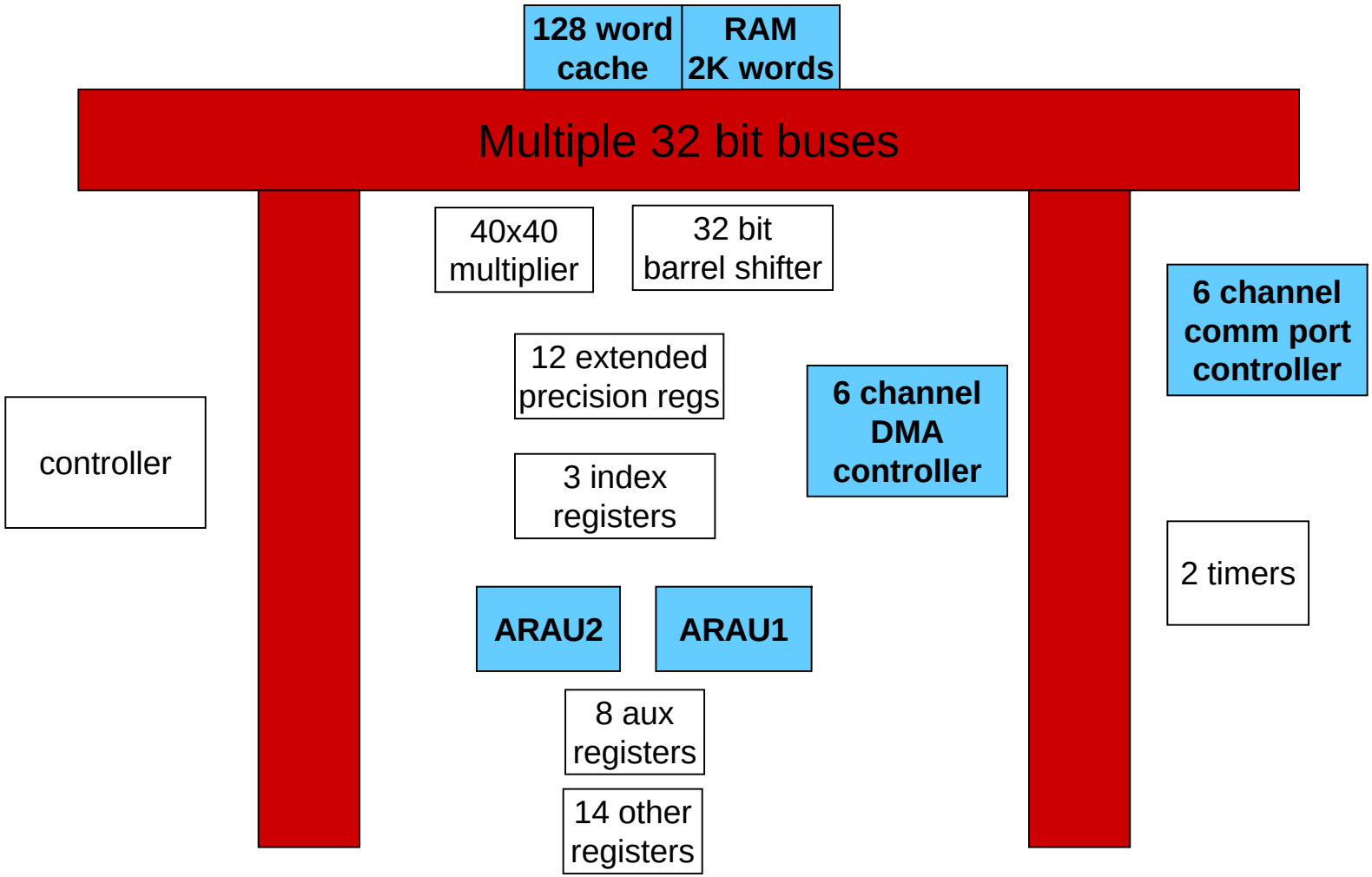
Review of Computer Architecture



Embedded System Architecture



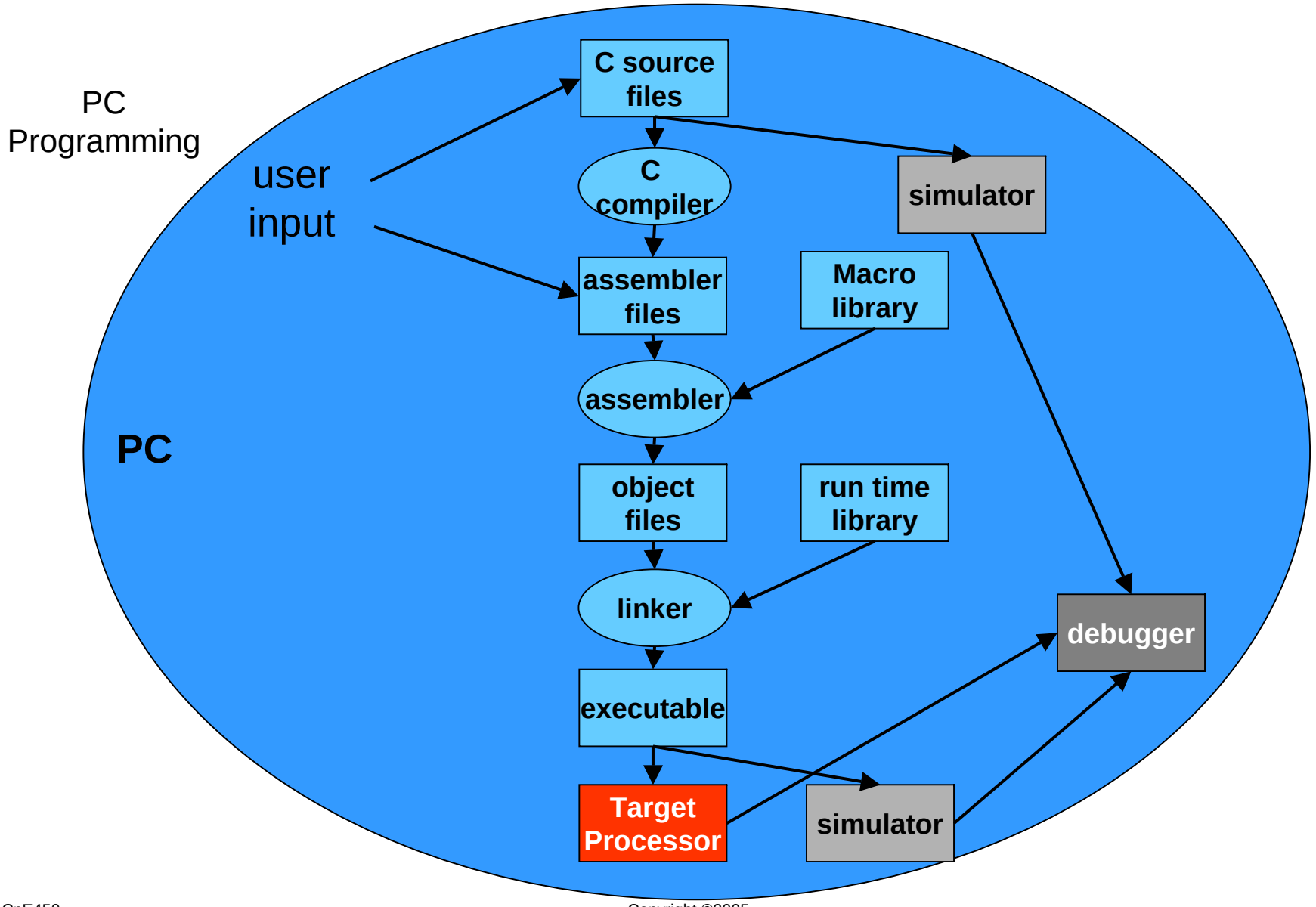
DSPs – the Ultimate Embedded Processors: TMS320C40 Architecture



Software Environments

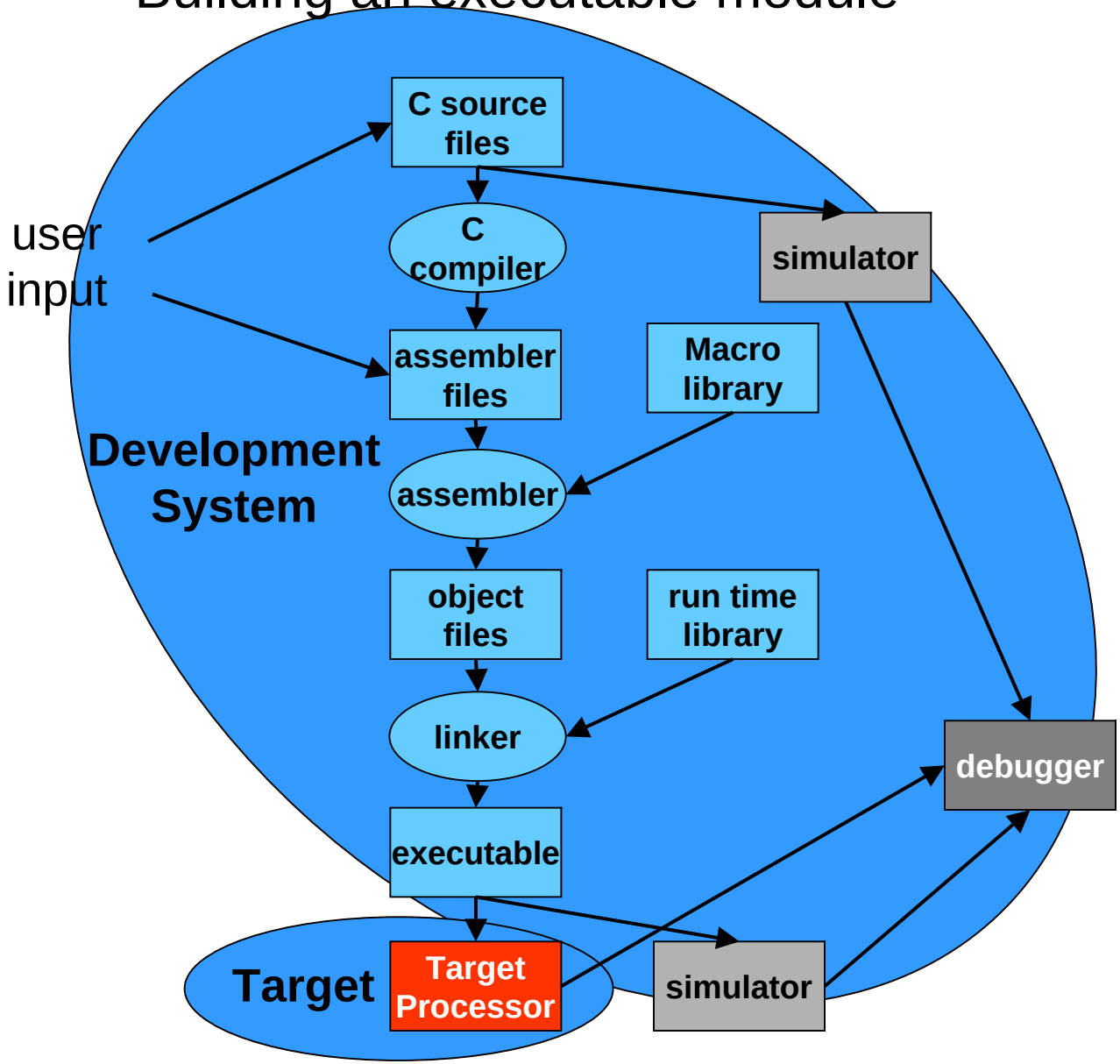
- Terminology:
  - Development system: Where the code for the end application is developed. Also known as development environment
  - Target system: Where the final code will actually run. Also known as execution environment

Building an executable module

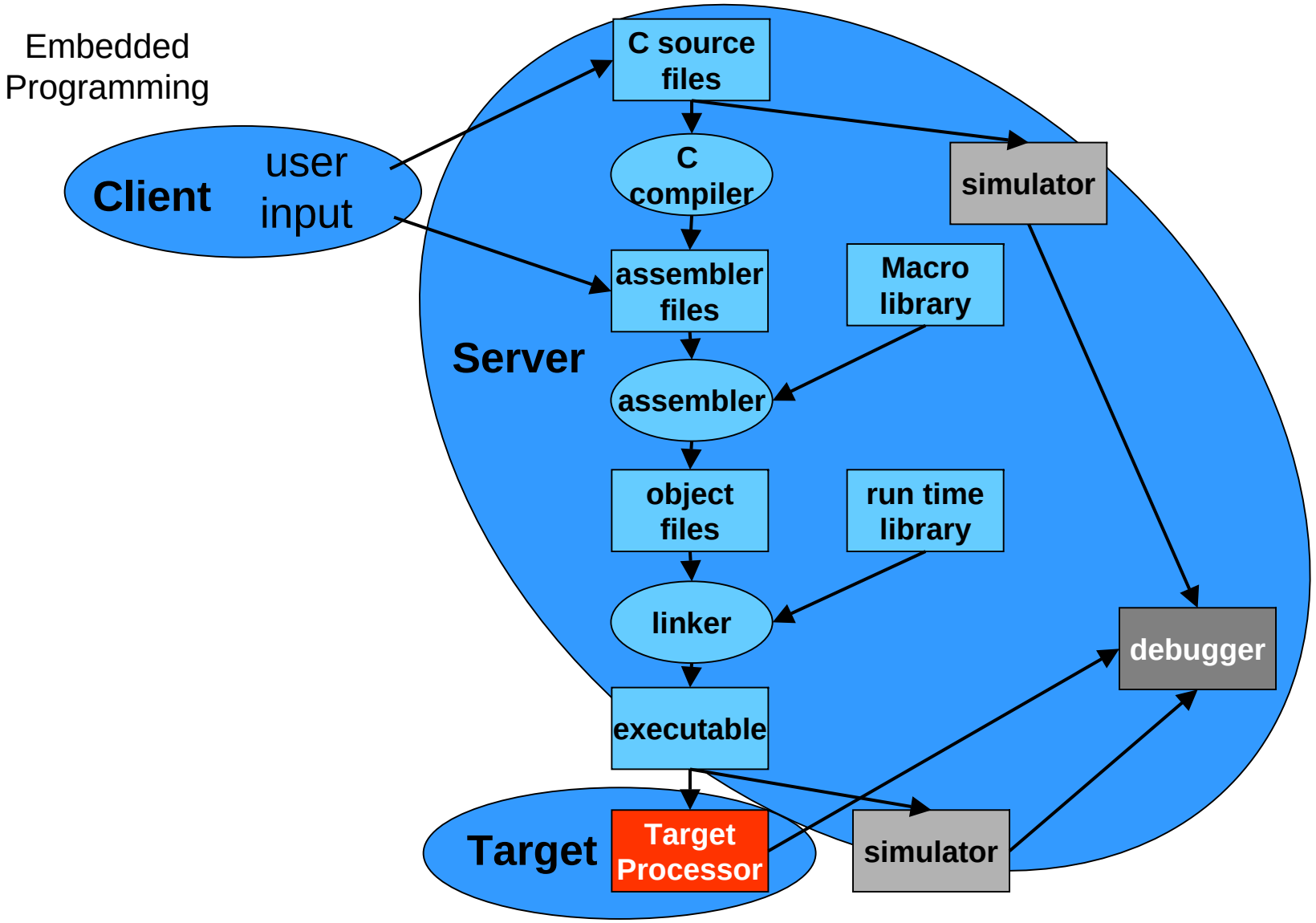


Building an executable module

Production Programming



Building an executable module



Typical Real-time System Constraints

- Input data is streaming at S samples/second. B samples must be processed in a batch within B/S seconds to keep up with the input stream
- An external input requires a decision within T seconds

Real-Time Design

- Two issues:
  - Predicting the computation time
  - Controlling/minimizing the computation time

Programming Real-Time Systems

```
float dot(float *f, float *g, int N)
  {
  float sum=0;
  int i;

  for(i=0;i<N;i++)
    sum += *f++ * *g++;

  return(sum);
  }
```

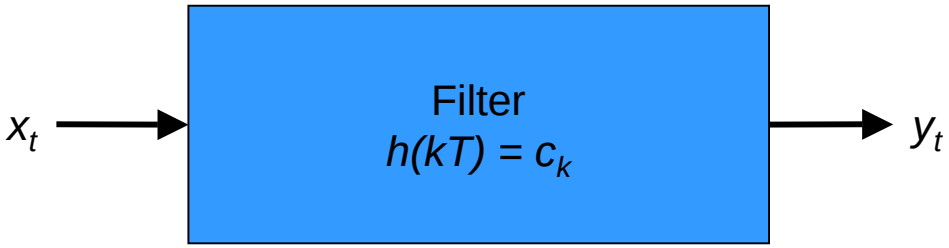
```
                                ;*****************************************************
                                ; FUNCTION DEF : _dot
                                ;*****************************************************
00000000                        _dot:
00000000 0f2b0000               PUSH    FP
00000001 080b0014               LDI     SP,FP
00000002 02740002               ADDI    2,SP
                        ;>>>>     float sum=0;
                        ;>>>>     int i;
00000003 07608000               LDF     0.0,R0
00000004 14400301               STF     R0,*+FP(1)
                        ;>>>>     for(i=0;i<N;i++)
00000005 08610000               LDI     0,R1
00000006 15410302               STI     R1,*+FP(2)
00000007 04c10b04               CMPI    *-FP(4),R1
00000008 6a0a000c               BGE     EPI0_1
00000009                        L1:
                        ;>>>>     sum += *f++ * *g++;
00000009 08480b02               LDI     *-FP(2),AR0
0000000a 08490b03               LDI     *-FP(3),AR1
0000000b 24e02021               MPYF    *AR1++,*AR0++,R0
0000000c 15480b02               STI     AR0,*-FP(2)
0000000d 15490b03               STI     AR1,*-FP(3)
0000000e 01c00301               ADDF    *+FP(1),R0
0000000f 14400301               STF     R0,*+FP(1)
00000010 08410302               LDI     *+FP(2),R1
00000011 02610001               ADDI    1,R1
00000012 15410302               STI     R1,*+FP(2)
00000013 04c10b04               CMPI    *-FP(4),R1
00000014 6a07fff4               BLT     L1
                        ;>>>>     return(sum);
00000015                        EPI0_1:
00000015 08410b01               LDI     *-FP(1),R1
00000016 68200001               BD      R1
00000017 084bc300               LDI     *FP,FP
00000018 0c800000               NOP
00000019 18740004               SUBI    4,SP
                        ***       B       R1      ;BRANCH OCCURS
```

```
                                ;*****************************************************
                                ; FUNCTION DEF : _dot
                                ;*****************************************************
00000000                        _dot:
00000000 0f2b0000               PUSH    FP
00000001 080b0014               LDI     SP,FP
00000002 0f2c0000               PUSH    AR4
                                ;
                                ; R2    assigned to variable  sum
                                ; AR2   assigned to parameter g
                                ; AR4   assigned to parameter f
                                ; RC    assigned to parameter N
                                ; RC    assigned to temp var  L$1
                                ;
00000003 1ecc0b02               LDA     *-FP(2),AR4
00000004 1eca0b03               LDA     *-FP(3),AR2
00000005 085b0b04               LDI     *-FP(4),RC
                        *** 5   ----------------------    sum = 0.0F;
00000006 07628000               LDF     0.0,R2
                        *** 8   ----------------------    if ( N <= 0 ) goto g4;
00000007 04fb0000               CMPI    0,RC
00000008 6a080005               BLE     L4
                        ***     ----------------------    L$1 = N-1;
00000009 371b1b01               SUBI    1,RC,RC
                        ***     ----------------------g3:
                        ***     ----------------------g5:
0000000a 07608000               LDF     0.0,R0
                        *** 9   ----------------------    sum += *f++**g++;
0000000b 139b001b               RPTS
0000000c 80102422               ADDF    R0,R2
                                ||  MPYF    *AR2++,*AR4++,R0
0000000d 01820000               ADDF    R0,R2
                        *** 8   ----------------------    if ( --L$1 >= 0 ) goto g5;
0000000e                        L4:
                        ***     ----------------------g4:
                        *** 11  ----------------------    return sum;
0000000e 07000002               LDF     R2,R0
0000000f                        EPI0_1:
0000000f 08410b01               LDI     *-FP(1),R1
00000010 68200001               BD      R1
00000011 084bc300               LDI     *FP,FP
00000012 0e2c0000               POP     AR4
00000013 18740002               SUBI    2,SP
                        ***     B       R1      ;BRANCH OCCURS
```

Assignment 2

- A common process one encounters in signal processing or control systems is the filtering operation. A filter can be represented as a coefficient array that is multiplied, sample by sample, with a set of data samples. Mathematically, this can be represented as:

$$y_t = \sum_{k=0}^{k=N} c_k \bullet x_{t-k}$$



1. What requirements (e.g., real-time, hardware architecture, memory, etc.) would this operation place on a embedded system being used for a filtering operation?
2. Write a C language routine (pseudo-code is OK) to illustrate how you might implement this function
3. How would you test the real-time performance of your implementation?