

Real-Time Embedded Systems

CpE-450 Spring 06

Class 1

Bruce McNair

bmcnair@stevens.edu

Course Introduction

- Logistics:
 - Instructor: Bruce McNair
 - Office: Burchard 206
 - Phone: 201-216-5549
 - email: bmcnair@stevens.edu
 - Office hours: Monday – Thursday, 9:30 – 4, subject to class and other meetings

Course Introduction

- Logistics:
 - Instructor: Bruce McNair
 - Office: Burchard 206
 - Phone: 201-216-5549
 - email: bmcnair@stevens.edu
 - Office hours: Monday – Thursday, 9:30 – 4, subject to class and other meetings
 - Homework
 - Must be typed or printed hardcopy, or electronic (i.e., *not* handwritten).
 - email is OK with MS/Office (2000 or previous), or program (e.g., .c, .m, ...) attachments. *Don't email me an executable or a macrovirus.*

Course Introduction

- Logistics:
 - Instructor: Bruce McNair
 - Office: Burchard 206
 - Phone: 201-216-5549
 - email: bmcnair@stevens.edu
 - Office hours: Monday – Thursday, 9:30 – 4, subject to class and other meetings
 - Homework
 - Must be typed or printed hardcopy, or electronic (i.e., *not* handwritten).
 - email is OK with MS/Office (2000 or previous), or program (e.g., .c, .m, ...) attachments. *Don't email me an executable or a macrovirus.*
 - **VERIFY THAT PROGRAMMING SUBMISSIONS INCLUDE ENOUGH ENVIRONMENT TO BE BUILT AND RUN** (e.g., .h files, initialization, etc.)
 - Include the problem statement with solution. Keep a copy of your hardcopy or electronic homework (it probably will not be returned)

Course Introduction

- Logistics:
 - Instructor: Bruce McNair
 - Office: Burchard 206
 - Phone: 201-216-5549
 - email: bmcnair@stevens.edu
 - Office hours: Monday – Thursday, 9:30 – 4, subject to class and other meetings
 - Homework
 - Must be typed or printed hardcopy, or electronic (i.e., *not* handwritten).
 - email is OK with MS/Office (2000 or previous), or program (e.g., .c, .m, ...) attachments. *Don't email me an executable or a macrovirus.*
 - **VERIFY THAT PROGRAMMING SUBMISSIONS INCLUDE ENOUGH ENVIRONMENT TO BE BUILT AND RUN** (e.g., .h files, initialization, etc.)
 - Include the problem statement with solution. Keep a copy of your hardcopy or electronic homework (it probably will not be returned)
 - **If you submit homework as an email attachment, make sure your name appears in the file. Don't assume that the email wrapper will accompany the attachment to the printer.**
 - **To ensure proper credit for the homework, indicate the due date on the homework**

Course Introduction

- Logistics:
 - Instructor: Bruce McNair
 - Office: Burchard 206
 - Phone: 201-216-5549
 - email: bmcnair@stevens.edu
 - Office hours: Monday – Thursday, 9:30 – 4, subject to class and other meetings
 - Homework
 - Must be typed or printed hardcopy, or electronic (i.e., *not* handwritten).
 - email is OK with MS/Office (2000 or previous), or program (e.g., .c, .m, ...) attachments. *Don't email me an executable or a macrovirus.*
 - **VERIFY THAT PROGRAMMING SUBMISSIONS INCLUDE ENOUGH ENVIRONMENT TO BE BUILT AND RUN** (e.g., .h files, initialization, etc.)
 - Include the problem statement with solution. Keep a copy of your hardcopy or electronic homework (it probably will not be returned)
 - **If you submit homework as an email attachment, make sure your name appears in the file. Don't assume that the email wrapper will accompany the attachment to the printer.**
 - **To ensure proper credit for the homework, indicate the due date on the homework**
 - Homework will be assigned every week. It is due 1 week later. My goal is to return it by the following class.
 - Problem solutions will be posted on-line - **LATE HOMEWORK WILL NOT BE ACCEPTED AFTER THE SOLUTION IS POSTED**

Course Introduction

- Logistics:
 - Instructor: Bruce McNair
 - Office: Burchard 206
 - Phone: 201-216-5549
 - email: bmcnair@stevens.edu
 - Office hours: Monday – Thursday, 9:30 – 4, subject to class and other meetings
 - Homework
 - Must be typed or printed hardcopy, or electronic (i.e., *not* handwritten).
 - email is OK with MS/Office (2000 or previous), or program (e.g., .c, .m, ...) attachments. *Don't email me an executable or a macrovirus.*
 - **VERIFY THAT PROGRAMMING SUBMISSIONS INCLUDE ENOUGH ENVIRONMENT TO BE BUILT AND RUN** (e.g., .h files, initialization, etc.)
 - Include the problem statement with solution. Keep a copy of your hardcopy or electronic homework (it probably will not be returned)
 - **If you submit homework as an email attachment, make sure your name appears in the file. Don't assume that the email wrapper will accompany the attachment to the printer.**
 - **To ensure proper credit for the homework, indicate the due date on the homework**
 - Homework will be assigned every week. It is due 1 week later. My goal is to return it by the following class.
 - Problem solutions will be posted on-line - **LATE HOMEWORK WILL NOT BE ACCEPTED AFTER THE SOLUTION IS POSTED**
 - **No assignment will require massive printout. Limit your results to a few pages**

Course Introduction

- Logistics:
 - Instructor: Bruce McNair
 - Office: Burchard 206
 - Phone: 201-216-5549
 - email: bmcnair@stevens.edu
 - Office hours: Monday – Thursday, 9:30 – 4, subject to class and other meetings
 - Homework
 - Must be typed or printed hardcopy, or electronic (i.e., *not* handwritten).
 - email is OK with MS/Office (2000 or previous), or program (e.g., .c, .m, ...) attachments. *Don't email me an executable or a macrovirus.*
 - **VERIFY THAT PROGRAMMING SUBMISSIONS INCLUDE ENOUGH ENVIRONMENT TO BE BUILT AND RUN** (e.g., .h files, initialization, etc.)
 - Include the problem statement with solution. Keep a copy of your hardcopy or electronic homework (it probably will not be returned)
 - **If you submit homework as an email attachment, make sure your name appears in the file. Don't assume that the email wrapper will accompany the attachment to the printer.**
 - **To ensure proper credit for the homework, indicate the due date on the homework**
 - Homework will be assigned every week. It is due 1 week later. My goal is to return it by the following class.
 - Problem solutions will be posted on-line - **LATE HOMEWORK WILL NOT BE ACCEPTED AFTER THE SOLUTION IS POSTED**
 - **No assignment will require massive printout. Limit your results to a few pages**
 - Grading – all submissions are to be an individual effort
 - Homework: 20%
 - Project: 30%
 - Midterm: 25%
 - Final: 25%
- Detailed grades will be posted on WebCT

Reference Materials

- Two optional texts, not required:
 - John Catsoulis, “Designing Embedded Hardware,” O’Reilly, 2003, ISBN 0-596-00362-5
 - Michael Barr, “Programming Embedded Systems in C and C++,” O’Reilly, 1999, ISBN 1-56592-354-5
- Reference material I’ll supply:
 - Circuit Cellar Magazine

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

Syllabus

6. Software structures
 - ISRs
 - Polling
 - Semaphore
7. Midterm
8. Evaluating system performance
 - Correctness
 - Speed
9. More on system performance evaluation
10. Profiling system performance
11. More on performance profiling
12. Performance optimization
 - Hand-optimization
 - Optimizing compilers
 - Pareto principle
13. Future directions, course project due
14. Final Exam

Definition of Terms

- Embedded System –

- Real-time Application –

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 –

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

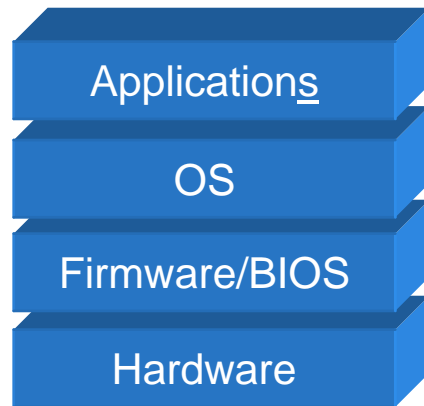
Typical Constraints on an Embedded System vs. Standalone System

Constraint	Embedded System	Standalone System
Cost	\$3	\$2000
Power	1 mW	500 W
Size	1 in ³	1 ft ³
Processing complexity	1-10 MIPS	500 – 1000 Mflops
Interfaces	8-16 bits	RS-232, Parallel, IDE, SCSI, USB, IEEE1394, WiFi, Bluetooth
O/S	Minimal or none	Windows XP
RAM	10 ³ – 10 ⁶ bytes	10 ⁸ – 10 ⁹ bytes
Programming	1-time load, programmed in advance	Flexible applications, user programmable, wide variety of languages

Typical Real-time System Constraints

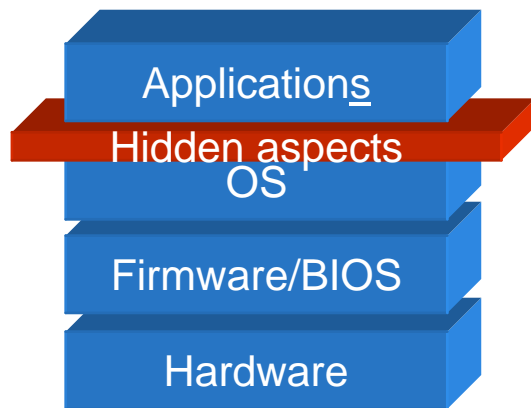
- Input data is streaming at S samples/second. B samples must be processed in a batch within B/S seconds to keep up with the input stream
- An external input requires a decision within T seconds

System Abstraction



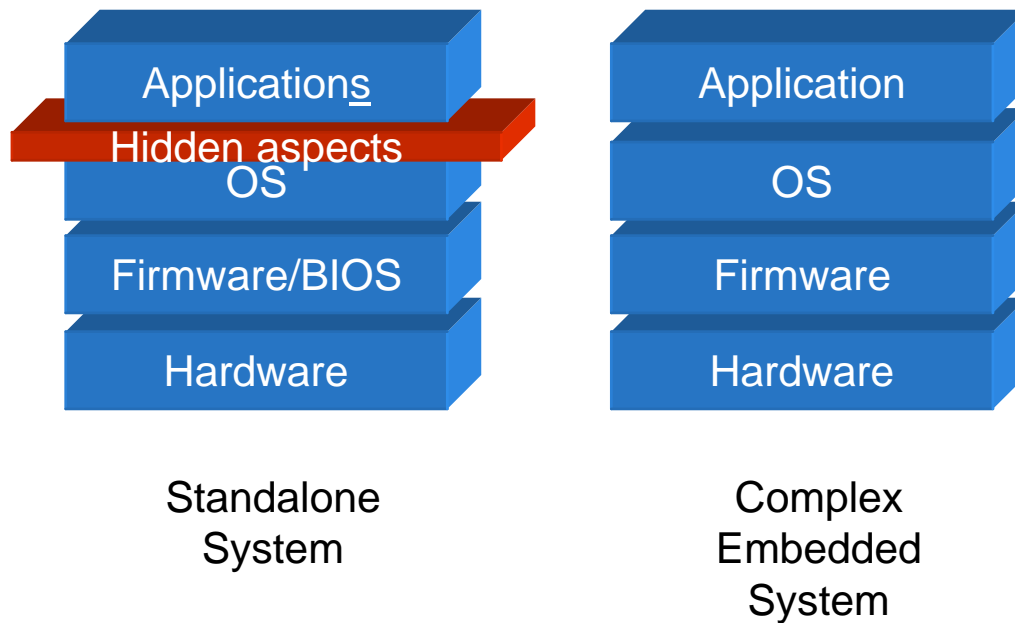
Standalone
System

System Abstraction

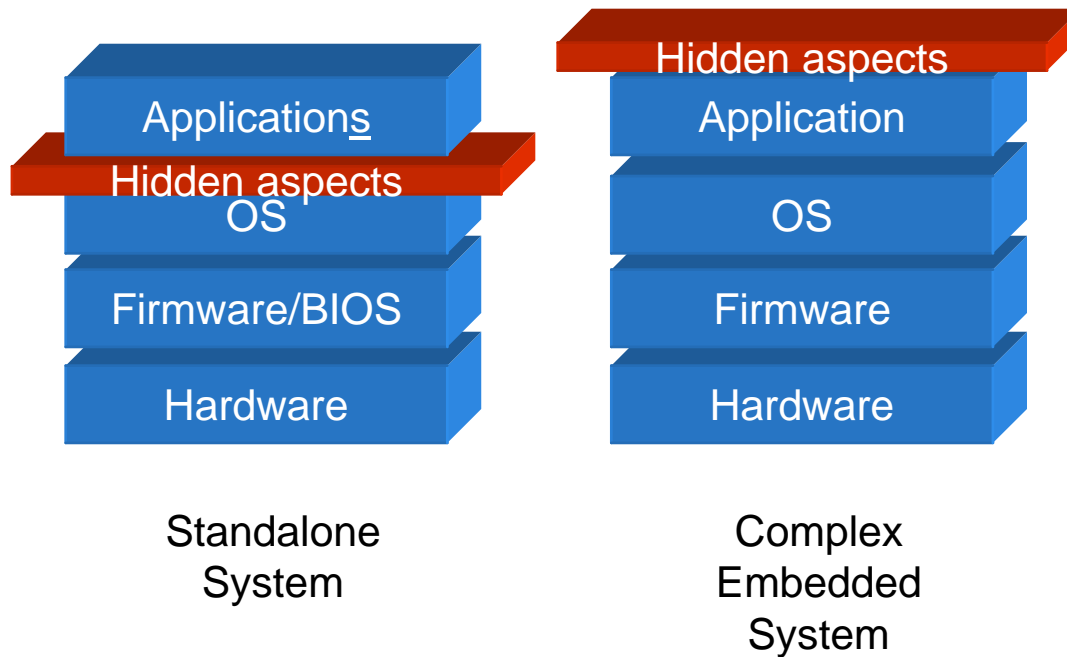


Standalone
System

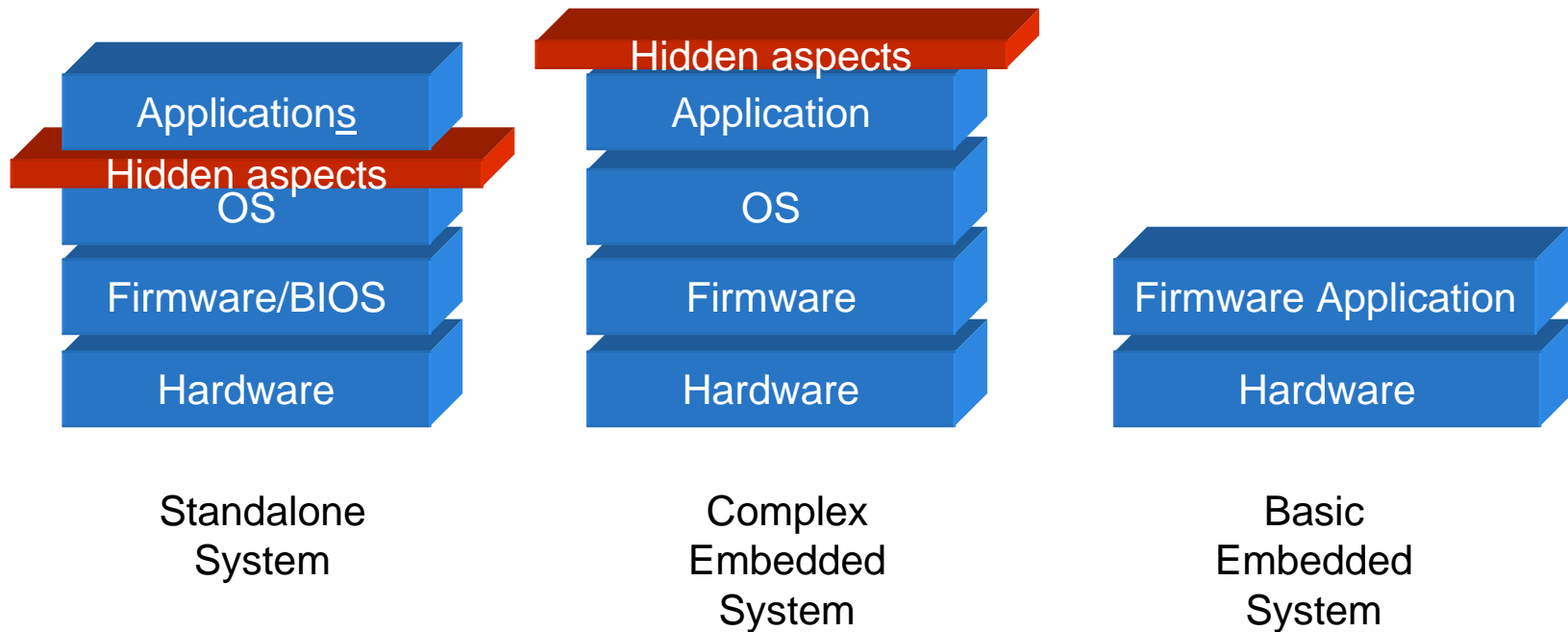
System Abstraction



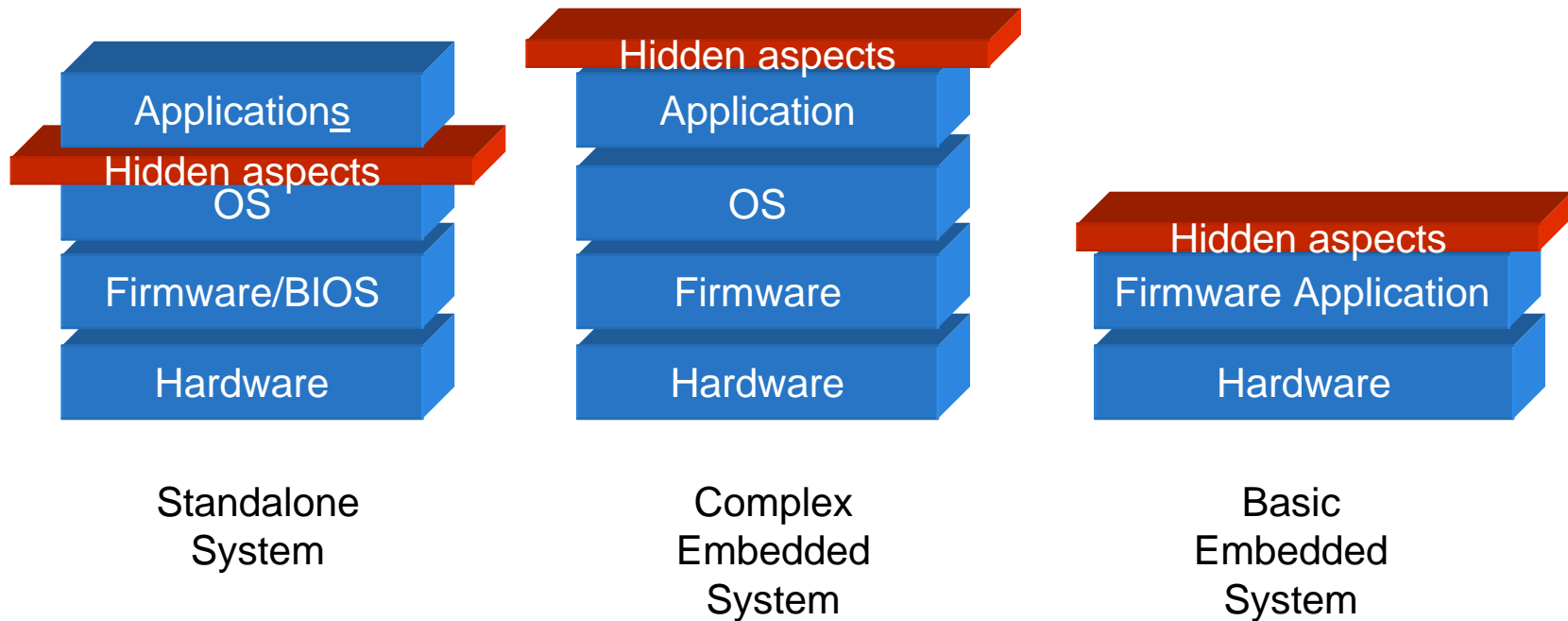
System Abstraction



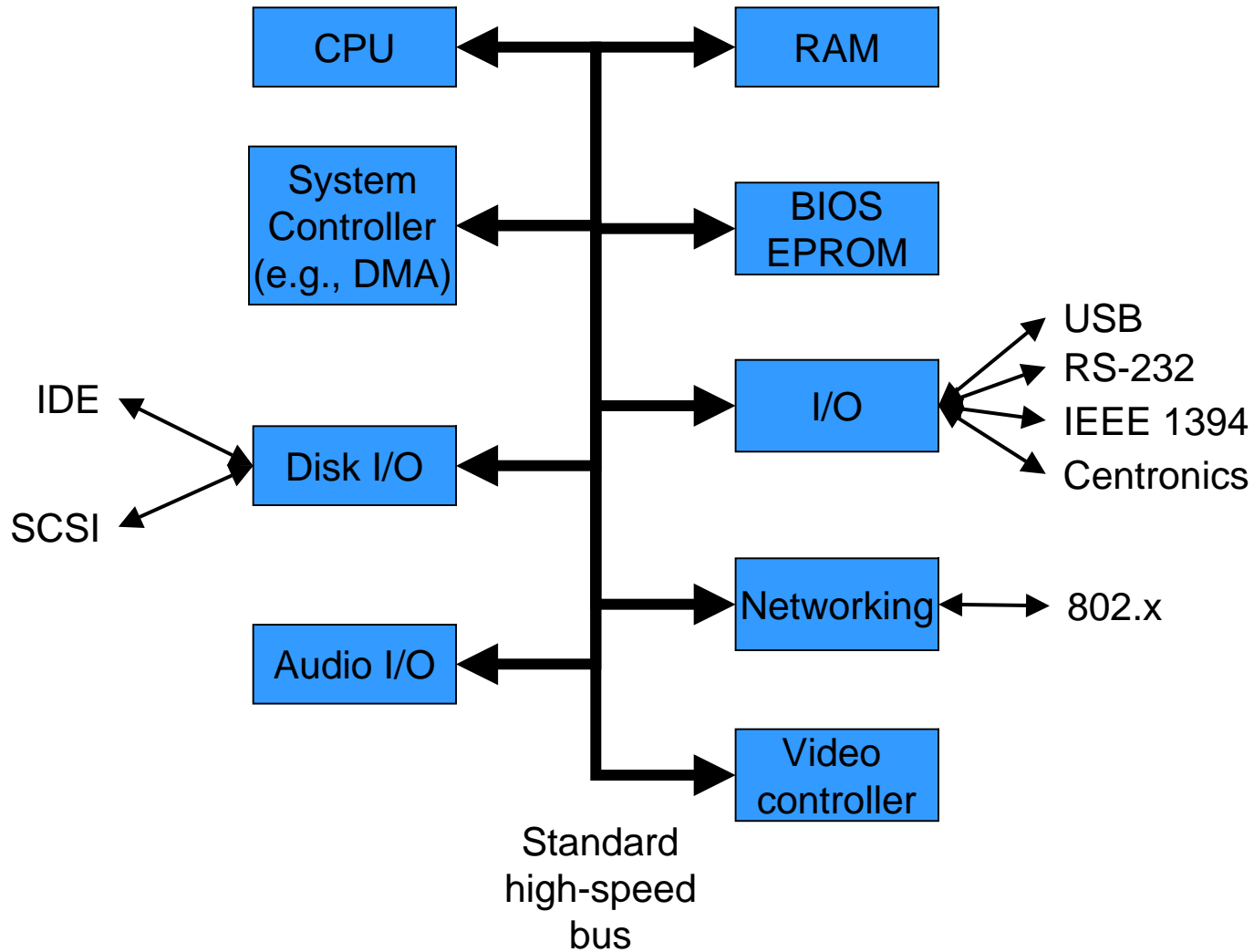
System Abstraction



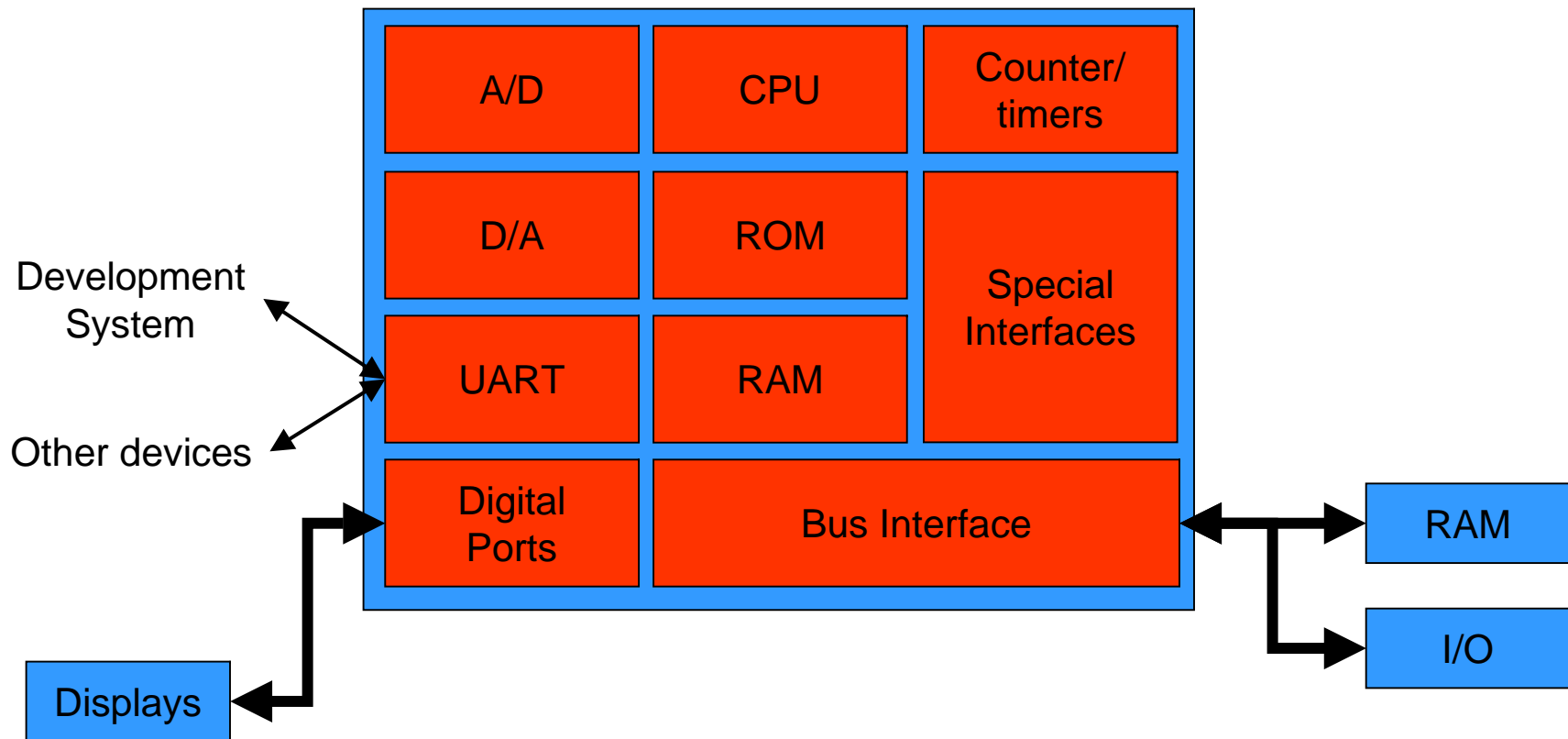
System Abstraction



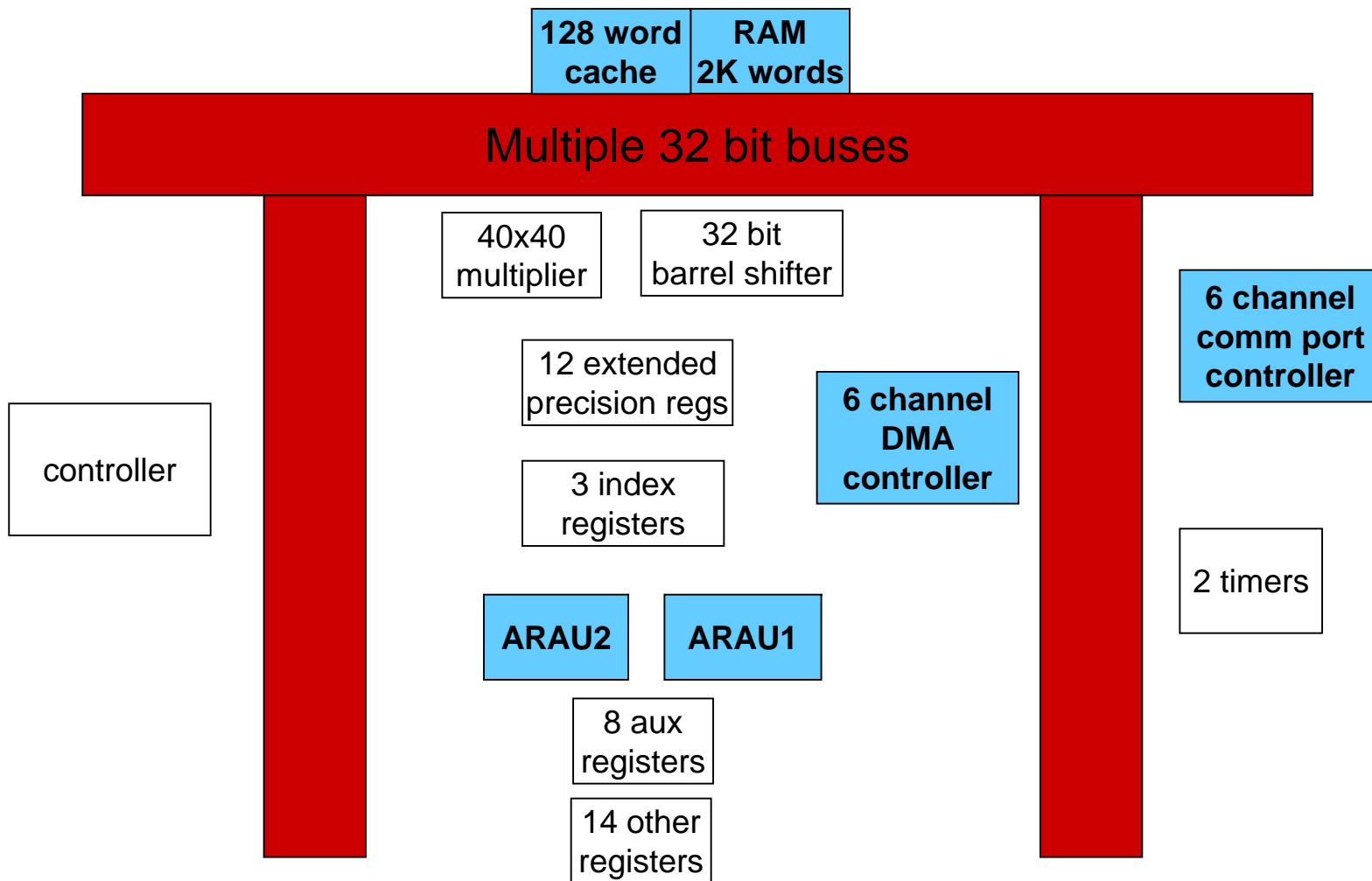
Review of Computer Architecture



Embedded System Architecture



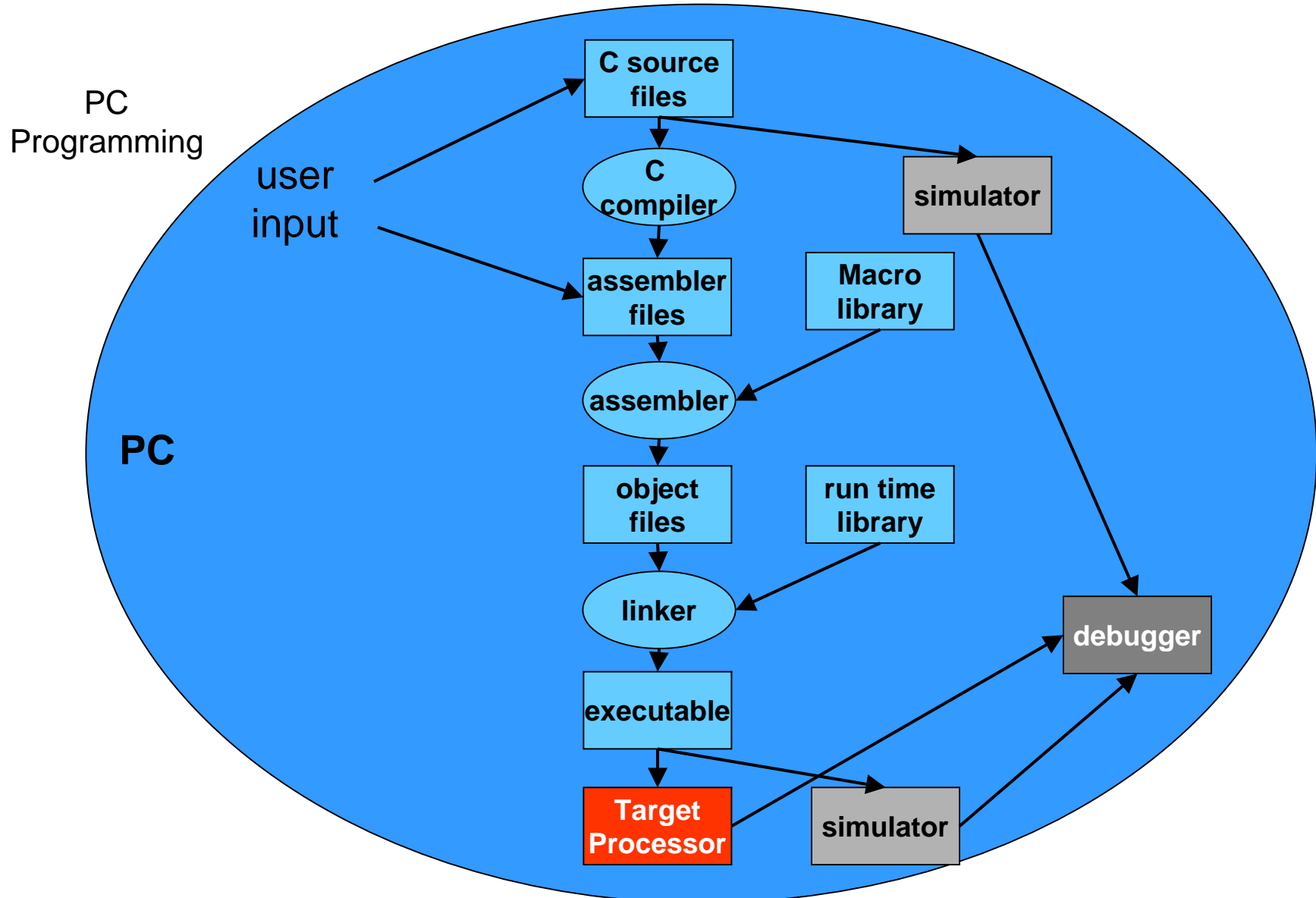
DSPs – the Ultimate Embedded Processors: TMS320C40 Architecture



Software Environments

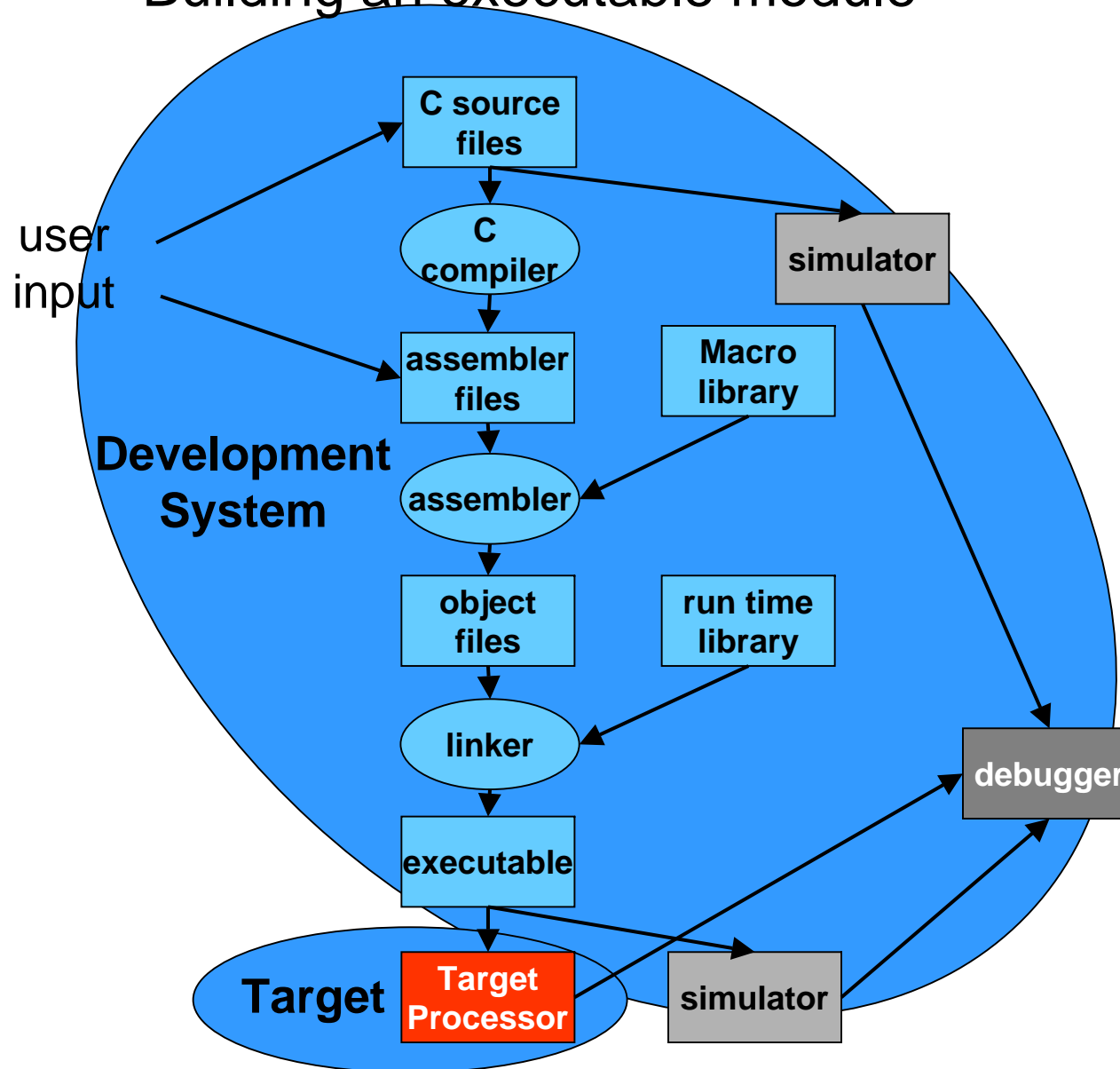
- Terminology:
 - Development system: Where the code for the end application is developed. Also known as development environment
 - Target system: Where the final code will actually run. Also known as execution environment

Building an executable module

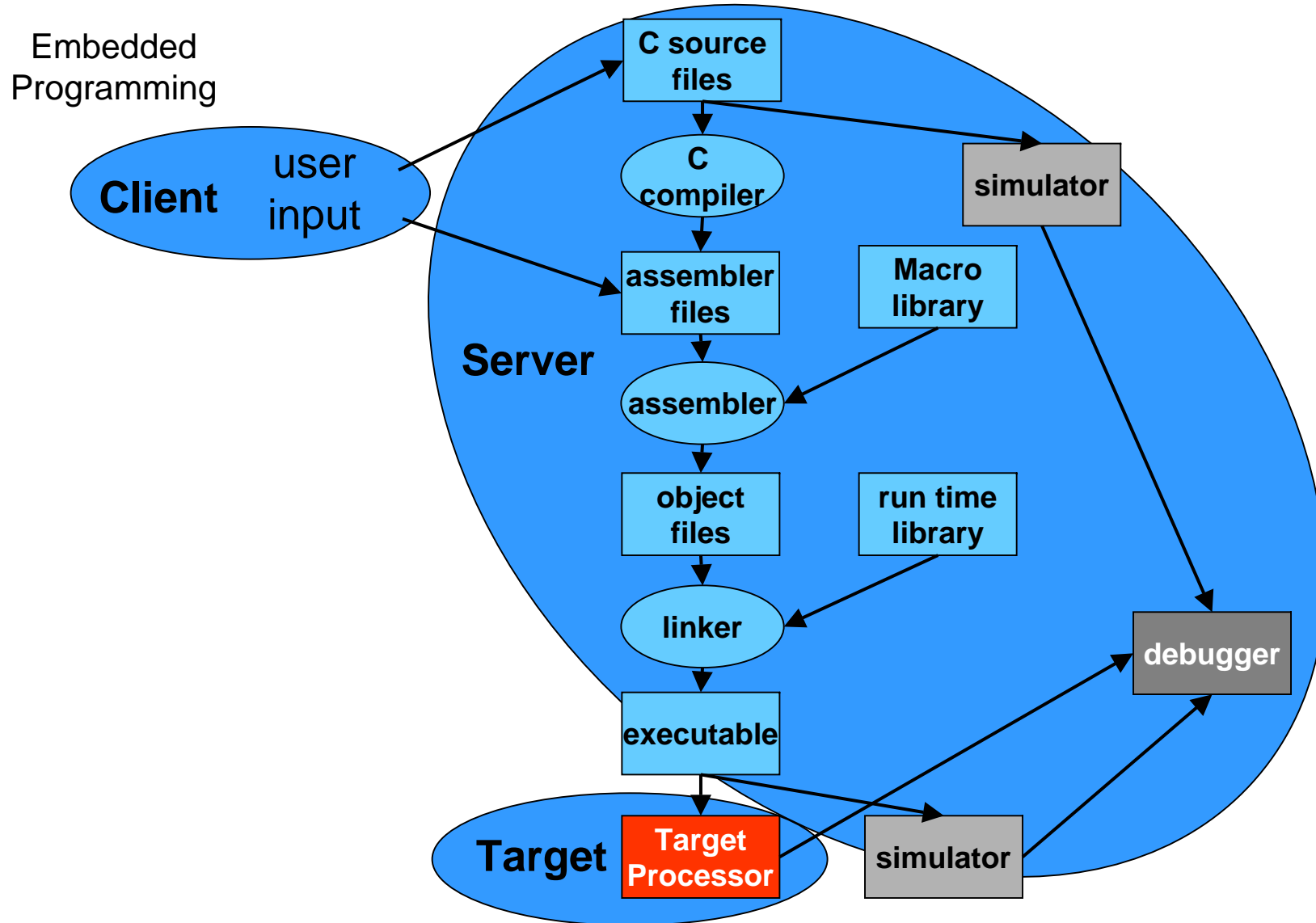


Building an executable module

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);
}
```

```

13 00000000      _dot:
14 00000000 0f2b0000    PUSH    FP
15 00000001 080b0014    LDI     SP,FP
16 00000002 02740002    ADDI    2,SP
17                ;>>>> float sum=0;
18                ;>>>> int i;
19 00000003 07608000    LDF    0.0,R0
20 00000004 14400301    STF    R0,*+FP(1)
21                ;>>>> for(i=0;i<N;i++)
22 00000005 08610000    LDI    0,R1
23 00000006 15410302    STI    R1,*+FP(2)
24 00000007 04c10b04    CMPI   *-FP(4),R1
25 00000008 6a0a000c    BGE    EPI0_1
26 00000009                L1:
27                ;>>>> sum += *f++ * *g++;
28 00000009 08480b02    LDI    *-FP(2),AR0
29 0000000a 08490b03    LDI    *-FP(3),AR1
30 0000000b 24e02021    MPYF   *AR1++,*AR0++,R0
31 0000000c 15480b02    STI    AR0,*-FP(2)
32 0000000d 15490b03    STI    AR1,*-FP(3)
33 0000000e 01c00301    ADDF   **FP(1),R0
34 0000000f 14400301    STF    R0,*+FP(1)
35 00000010 08410302    LDI    **FP(2),R1
36 00000011 02610001    ADDI   1,R1
37 00000012 15410302    STI    R1,*+FP(2)
38 00000013 04c10b04    CMPI   *-FP(4),R1
39 00000014 6a07fff4    BLT    L1
40                ;>>>> return(sum);
41 00000015      EPI0_1:
42 00000015 08410b01    LDI    *-FP(1),R1
43 00000016 68200001    BD     R1
44 00000017 084bc300    LDI    *FP,FP
45 00000018 0c800000    NOP
46 00000019 18740004    SUBI   4,SP
47                *** B R1 ;BRANCH OCCURS
```

```

98                *****
99                * FUNCTION DEF : _dot
100               *****
101 00000000      _dot:
102 00000000 0f2b0000    PUSH    FP
103 00000001 080b0014    LDI     SP,FP
104 00000002 0f2c0000    PUSH    AR4
105                *
106                * R2 assigned to variable sum
107                * AR2 assigned to parameter g
108                * AR4 assigned to parameter f
109                * RC assigned to parameter N
110                * RC assigned to temp var L$1
111                *
112 00000003 1ecc0b02    LDA    *-FP(2),AR4
113 00000004 1eca0b03    LDA    *-FP(3),AR2
114 00000005 085b0b04    LDI    *-FP(4),RC
115                *** 5 ----- sum = 0.0F;
116 00000006 07628000    LDF    0.0,R2
117                *** 8 ----- if ( N <= 0 ) goto g4;
118 00000007 04fb0000    CMPI   0,RC
119 00000008 6a080005    BLE    L4
120                *** ----- L$1 = N-1;
121 00000009 371b1b01    SUBI   1,RC,RC
122                *** -----g3:
123                *** -----g5:
124 0000000a 07608000    LDF    0.0,R0
125                *** 9 ----- sum += *f++ * *g++;
126 0000000b 139b001b    RPTS
127 0000000c 80102422    ADDF   R0,R2
128                || MPYF *AR2++,*AR4++,R0
129 0000000d 01820000    ADDF   R0,R2
130                ** 8 ----- if ( --L$1 >= 0 ) goto g5;
131 0000000e      L4:
132                *** -----g4:
133                *** 11 ----- return sum;
134 0000000e 07000002    LDF    R2,R0
135 0000000f      EPI0_1:
136 0000000f 08410b01    LDI    *-FP(1),R1
137 00000010 68200001    BD     R1
138 00000011 084bc300    LDI    *FP,FP
139 00000012 0e2c0000    POP    AR4
140 00000013 18740002    SUBI   2,SP
141                *** B R1 ;BRANCH OCCURS
```

Assignment 1

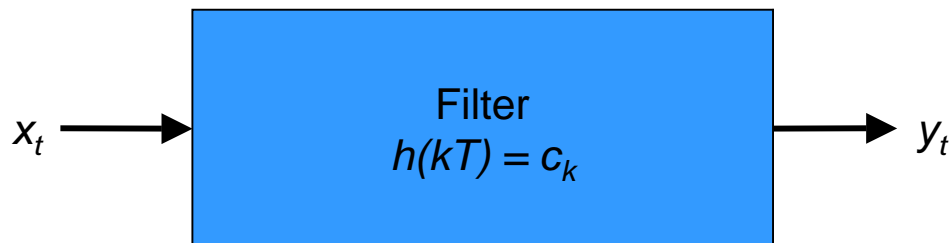
1. Identify at least 3 embedded systems that you might encounter in your room, your car or some other environment that you interact with on a regular basis.
 - Identify whether these systems have any real-time performance requirements. If they do, identify the real-time requirements that exist for the system. What might happen if the real-time requirements were not met?
 - Pick one system and research how it is or has been typically implemented.

Embedded system	Are there any real-time requirements?	What are they?	Implications of failure to meet real-time requirements
1.			
2.			
3.			

Assignment 1

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:

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



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