Chap 13 - Slides

Stallings 9e

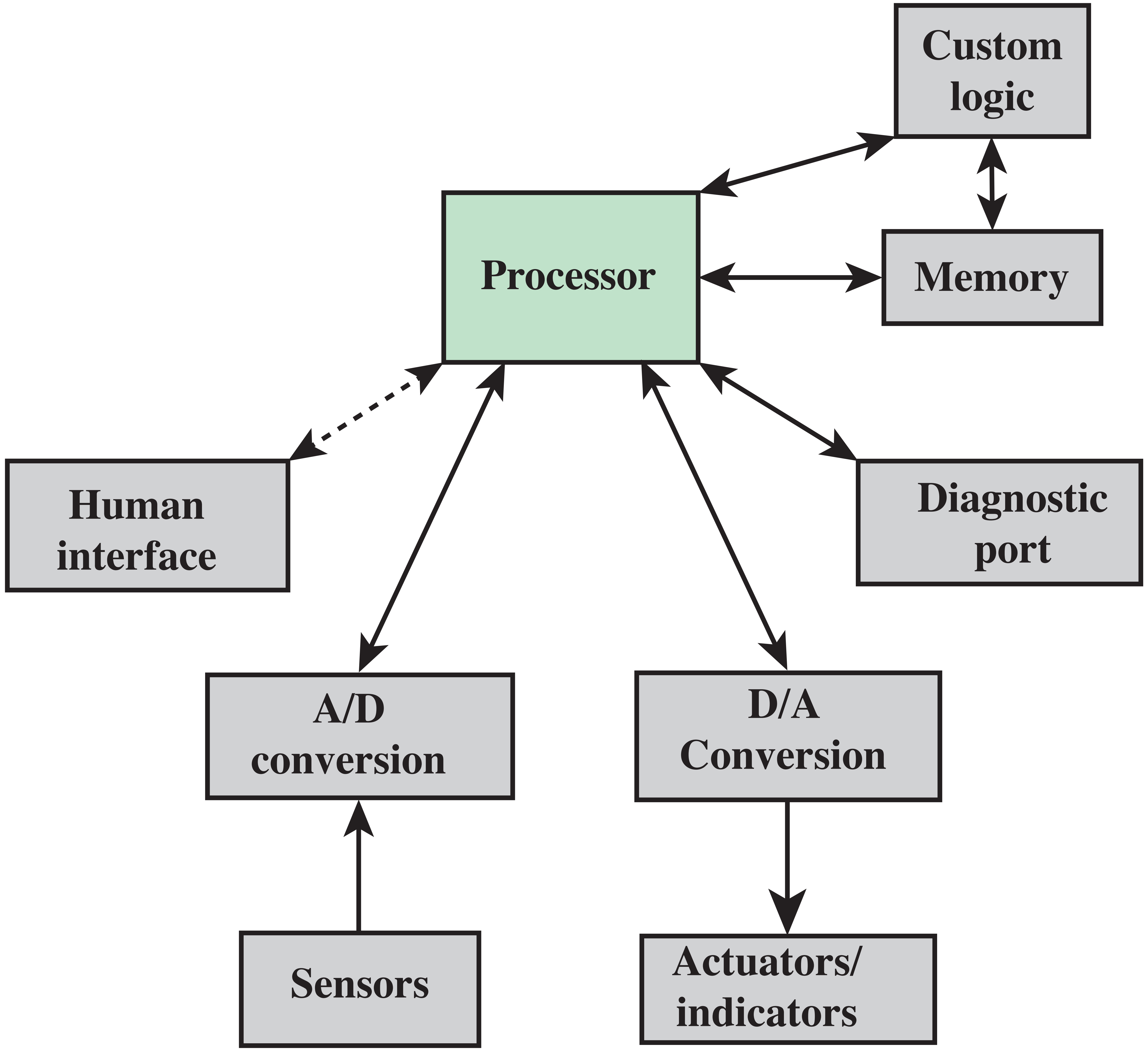


Figure 13.1 Possible Organization of an Embedded System

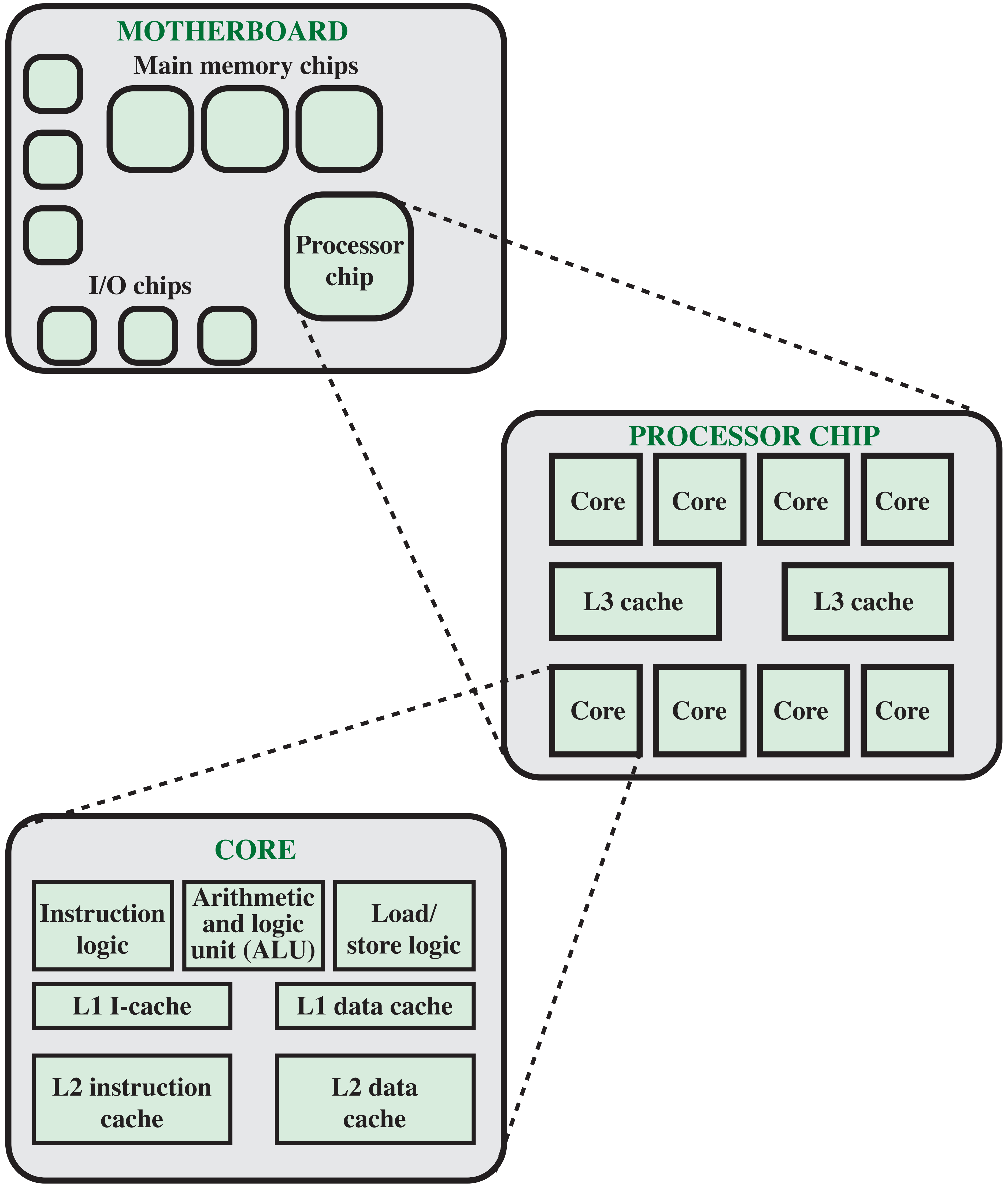
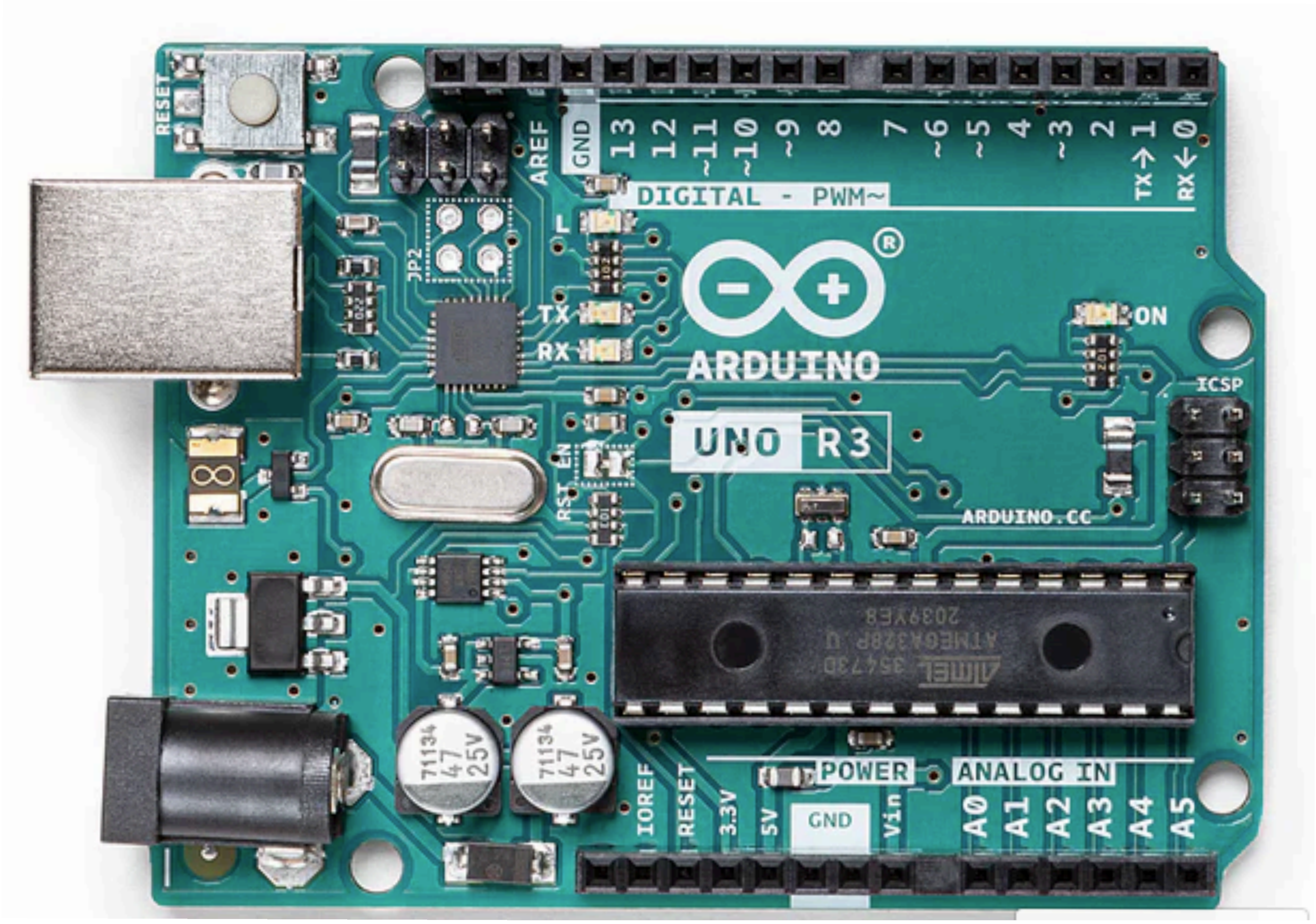


Figure 13.2 Simplified View of Major Elements of a Multicore Microcomputer

Arduino Uno


RESET
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL - PWM~
L
TX
RX
ARDUINO
UNO R3
ON
ICSP
ARDUINO.CC
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
Vin
A0
A1
A2
A3
A4
A5

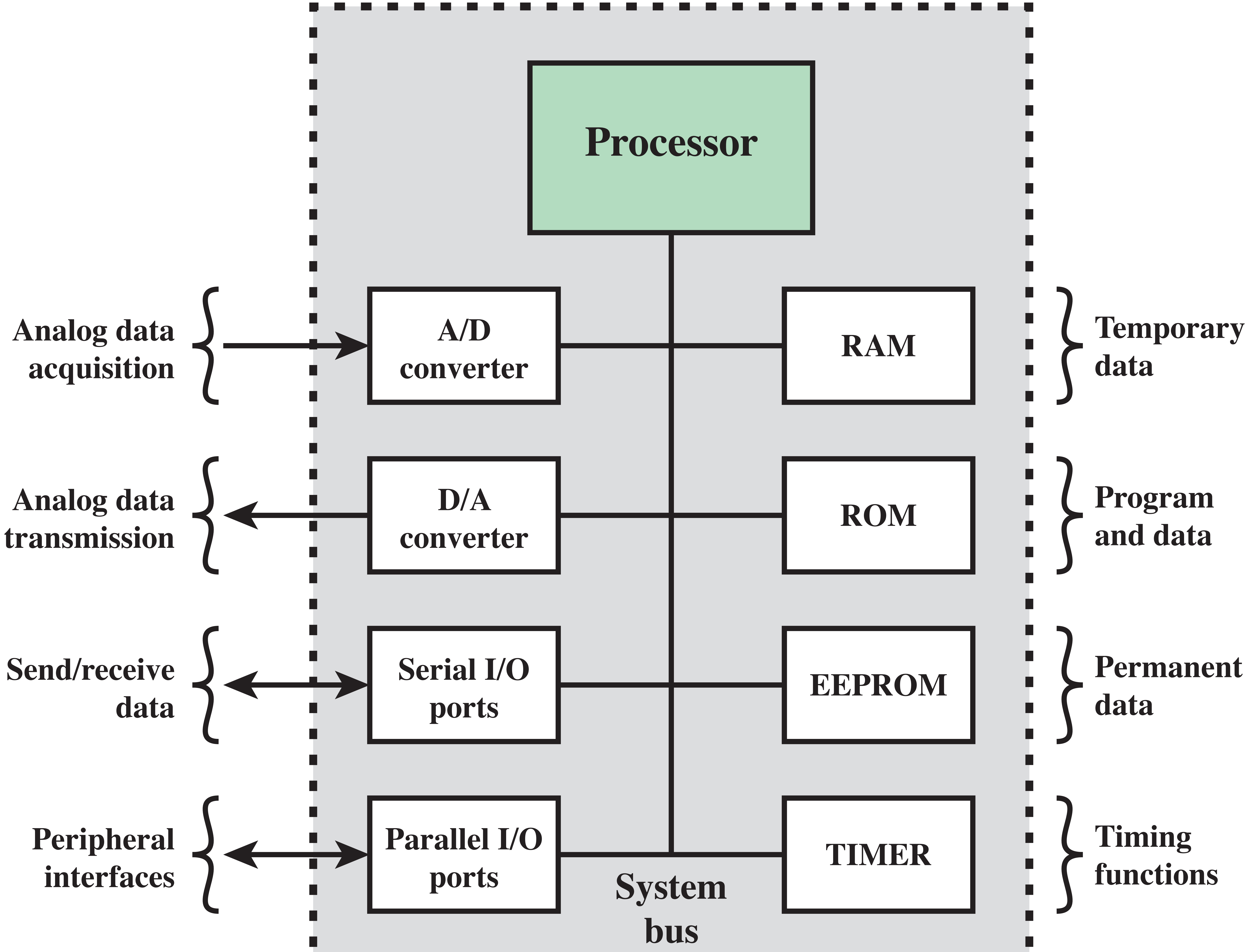
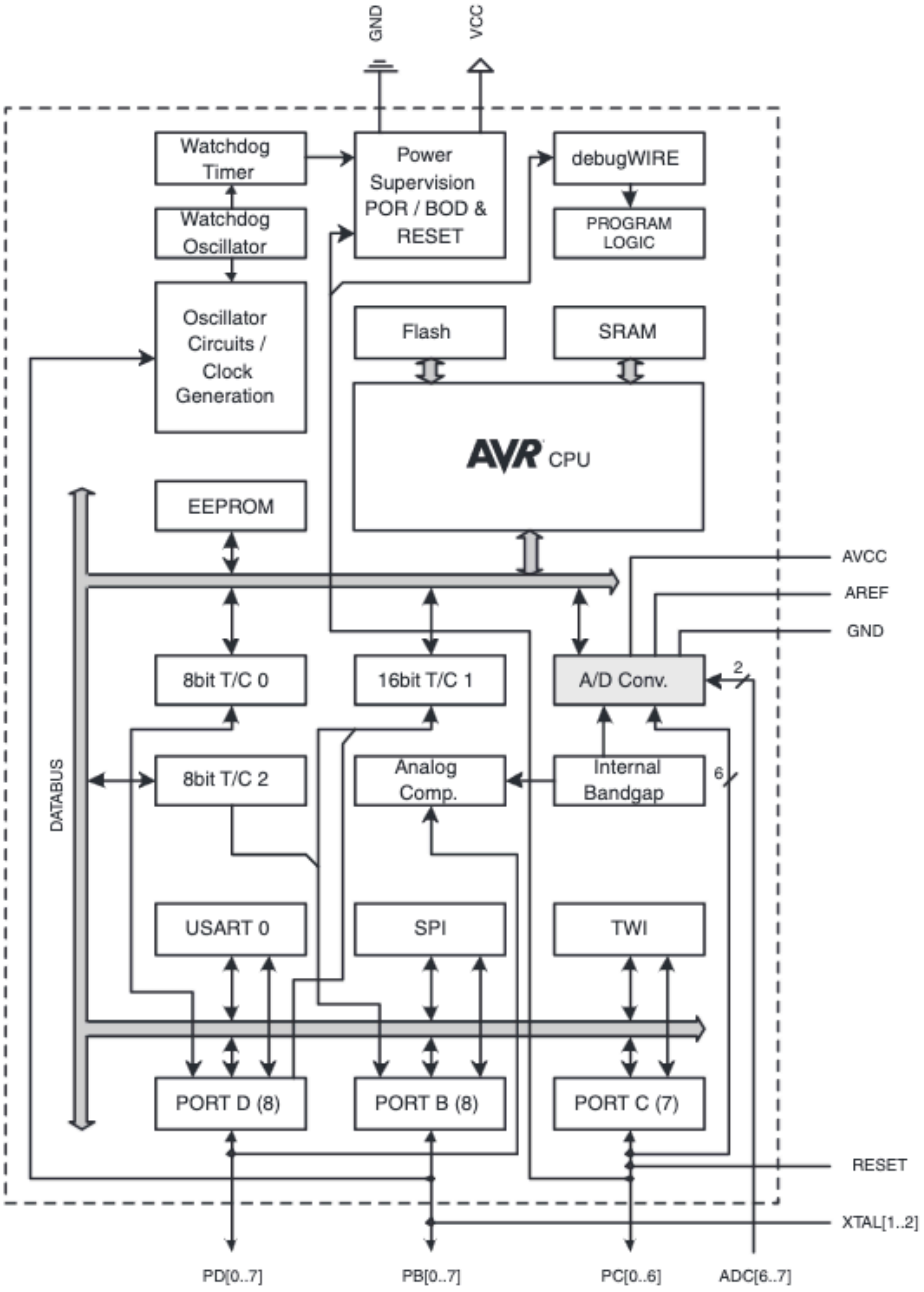


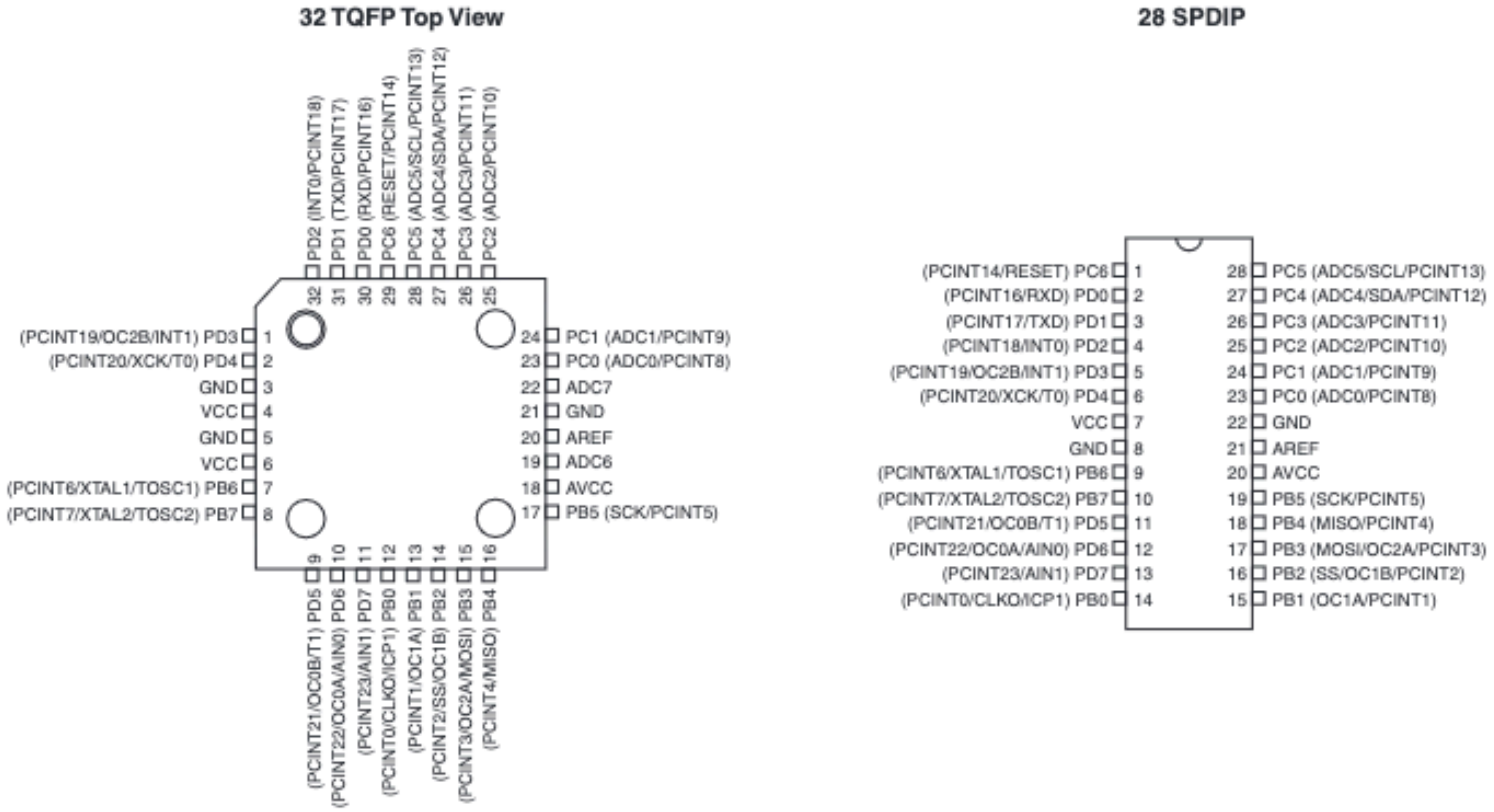
Figure 13.3 Typical Microcontroller Chip Elements

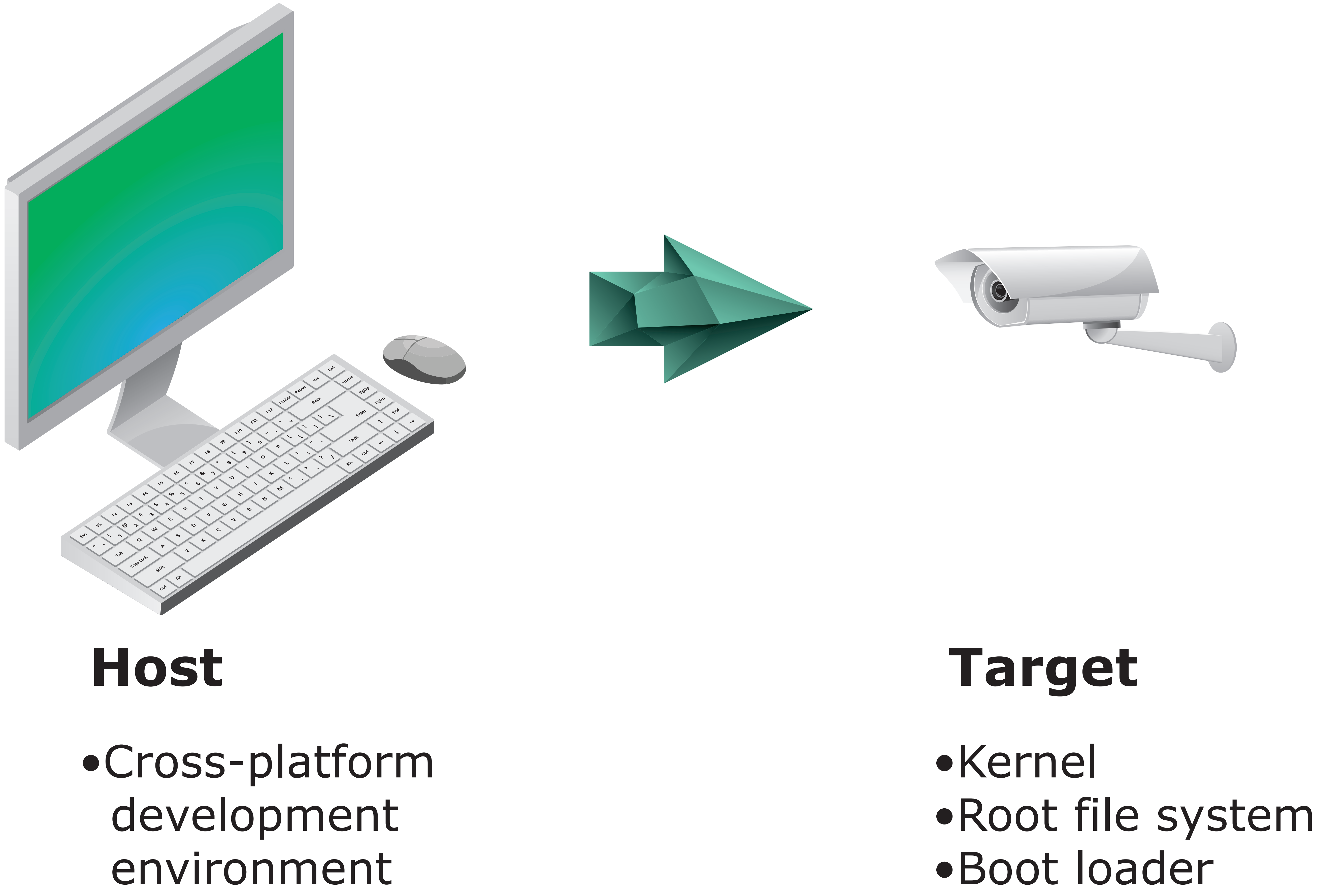
Atmega 328


GND
VCC
Watchdog Timer
Power Supervision POR / BOD & RESET
debugWIRE
Watchdog Oscillator
PROGRAM LOGIC
Oscillator Circuits / Clock Generation
Flash
SRAM
AVR CPU
EEPROM
AVCC
AREF
GND
8bit T/C 0
16bit T/C 1
A/D Conv.
2
DATABUS
8bit T/C 2
Analog Comp.
Internal Bandgap
6
USART 0
SPI
TWI
PORT D (8)
PORT B (8)
PORT C (7)
RESET
XTAL[1..2]
PD[0..7]
PB[0..7]
PC[0..6]
ADC[6..7]

Atmega 328 Pinout

Pinout ATmega48A/PA/88A/PA/168A/PA/328/P





Host

- Cross-platform development environment

Target

- Kernel
- Root file system
- Boot loader

Figure 13.4 Host-Target Environment

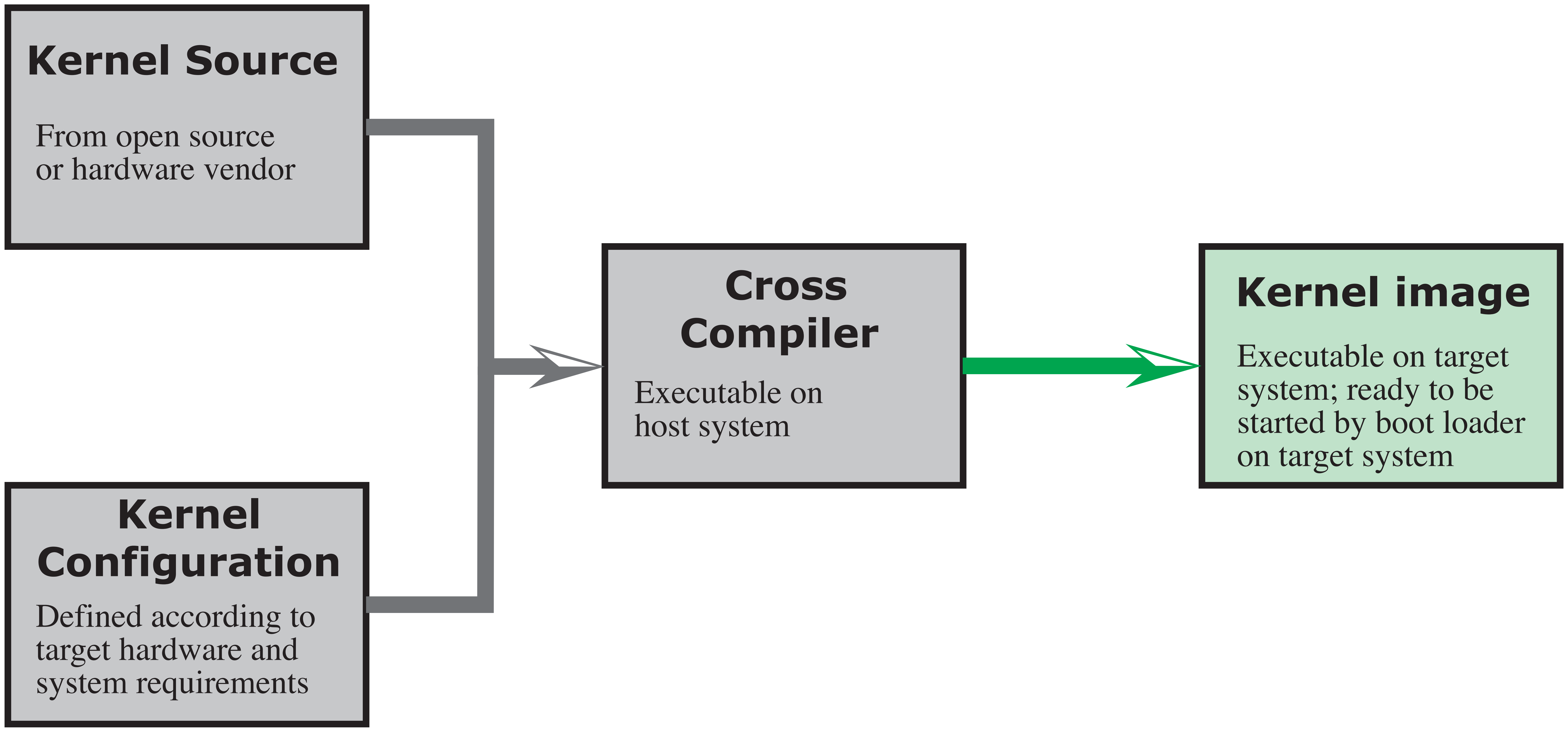

Kernel Source
From open source or hardware vendor
Kernel Configuration
Defined according to target hardware and system requirements
Cross Compiler
Executable on host system
Kernel image
Executable on target system; ready to be started by boot loader on target system

Figure 13.5 Kernel Compilation

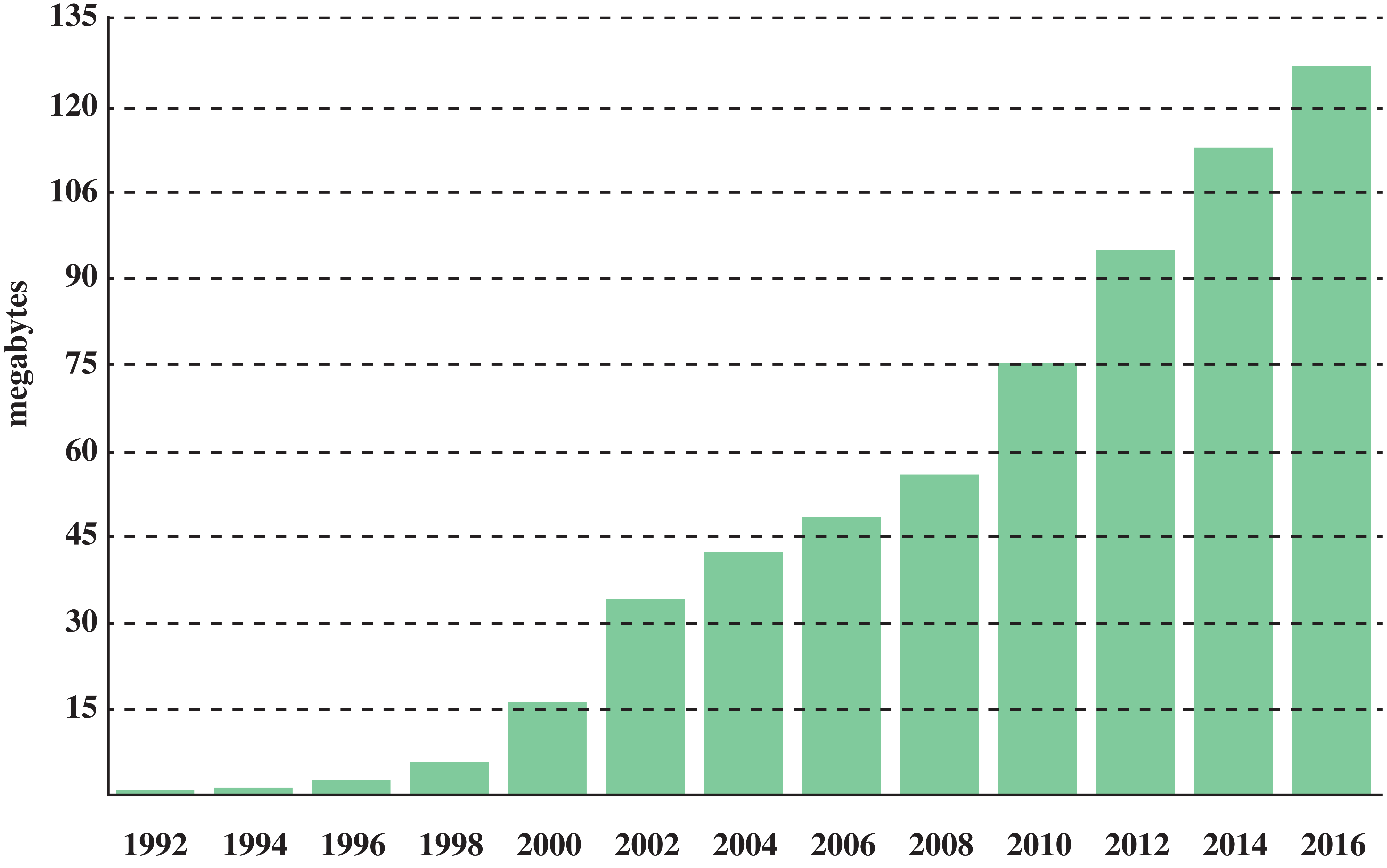
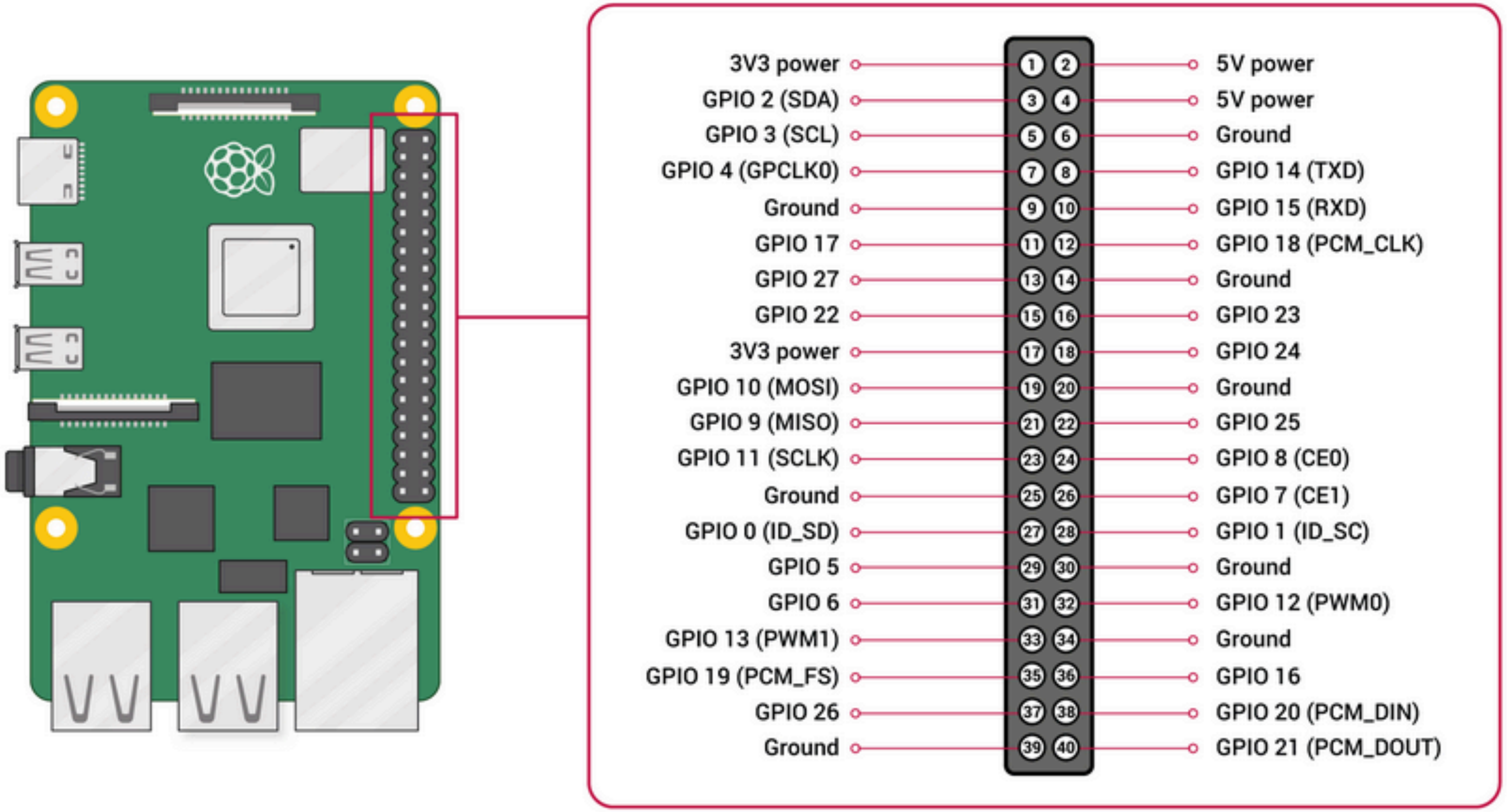


Figure 13.6 Size of Linux Kernel (shown in GZIP-compressed file size)

Raspberry Pi



3V3 power	1	2	5V power
GPIO 2 (SDA)	3	4	5V power
GPIO 3 (SCL)	5	6	Ground
GPIO 4 (GPCLK0)	7	8	GPIO 14 (TXD)
Ground	9	10	GPIO 15 (RXD)
GPIO 17	11	12	GPIO 18 (PCM_CLK)
GPIO 27	13	14	Ground
GPIO 22	15	16	GPIO 23
3V3 power	17	18	GPIO 24
GPIO 10 (MOSI)	19	20	Ground
GPIO 9 (MISO)	21	22	GPIO 25
GPIO 11 (SCLK)	23	24	GPIO 8 (CE0)
Ground	25	26	GPIO 7 (CE1)
GPIO 0 (ID_SD)	27	28	GPIO 1 (ID_SC)
GPIO 5	29	30	Ground
GPIO 6	31	32	GPIO 12 (PWM0)
GPIO 13 (PWM1)	33	34	Ground
GPIO 19 (PCM_FS)	35	36	GPIO 16
GPIO 26	37	38	GPIO 20 (PCM_DIN)
Ground	39	40	GPIO 21 (PCM_DOUT)

ARM Processor
512 MB Memory
80 MHz Clock

SoC - System on a Chip

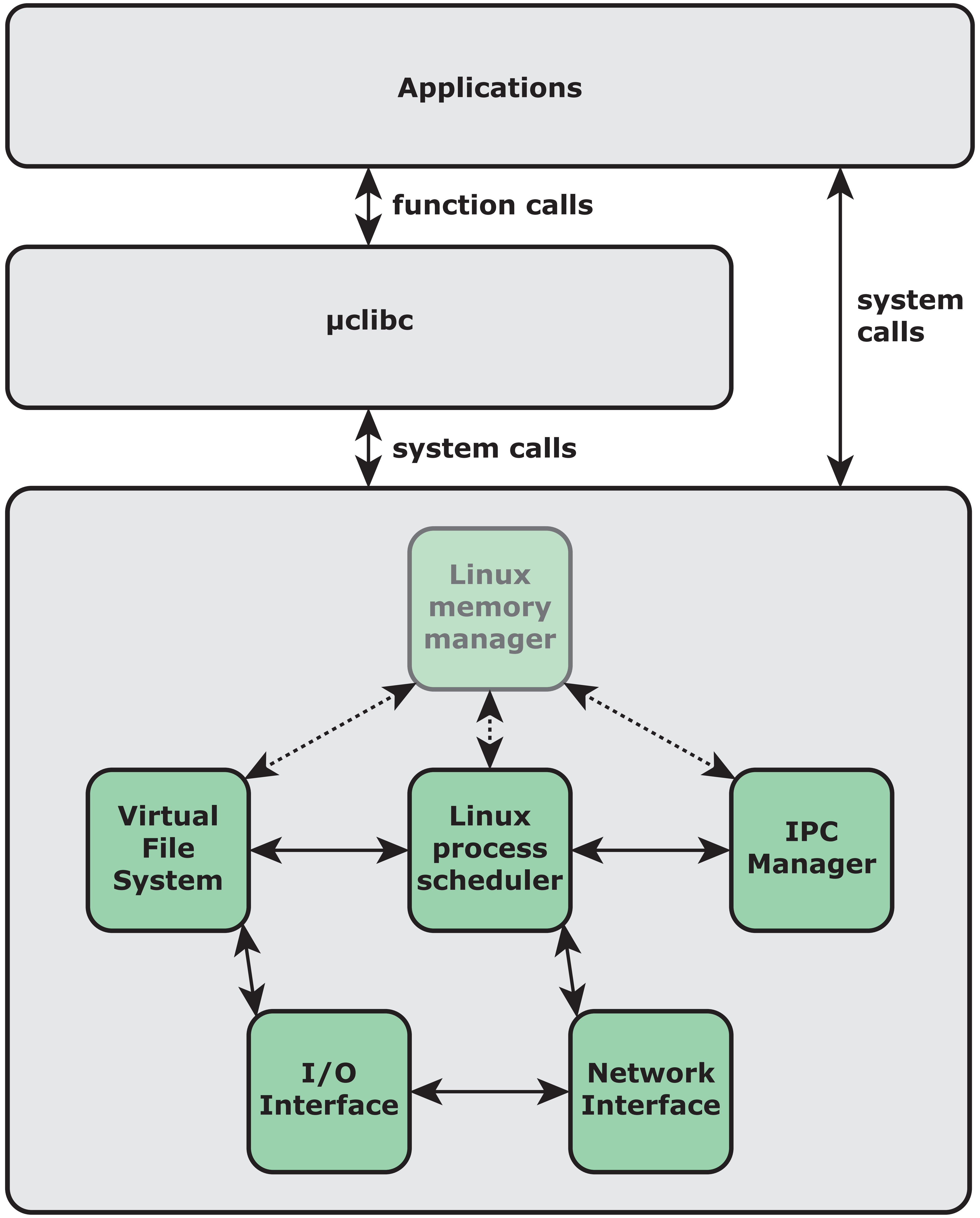
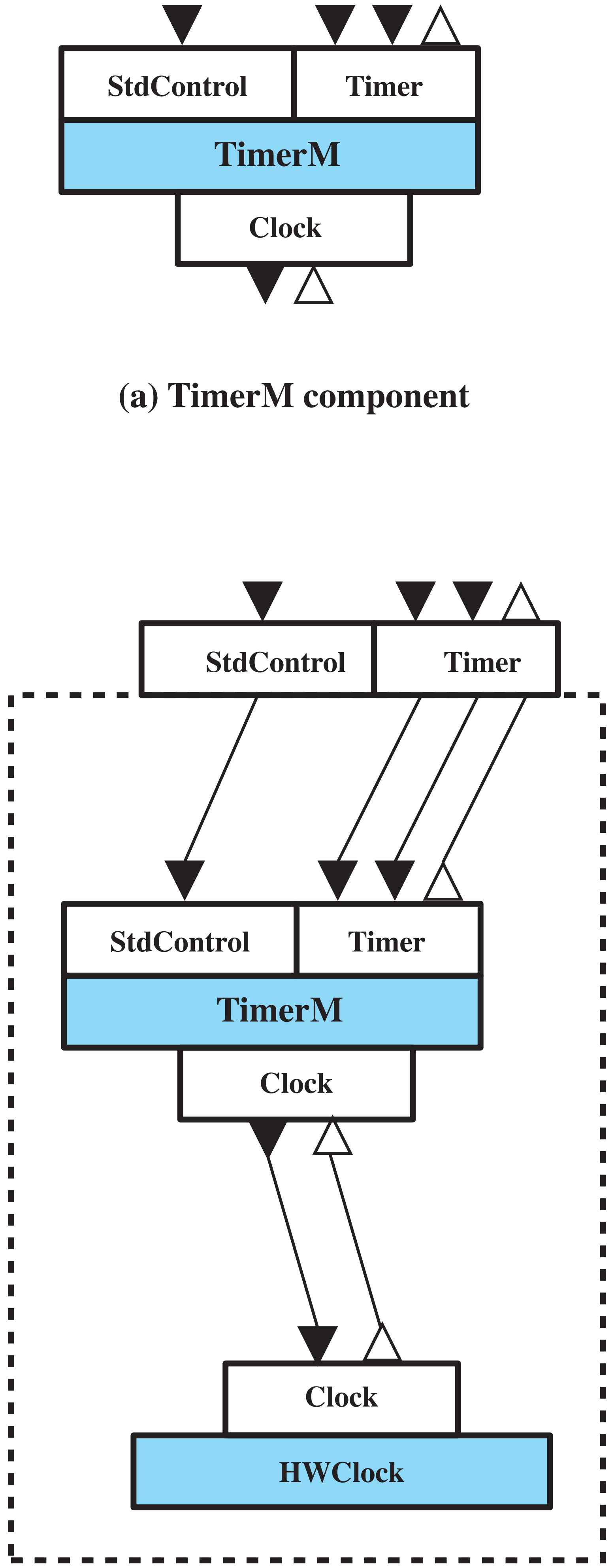


Figure 13.7 μcLinux/μclibc Software Architecture



(a) TimerM component

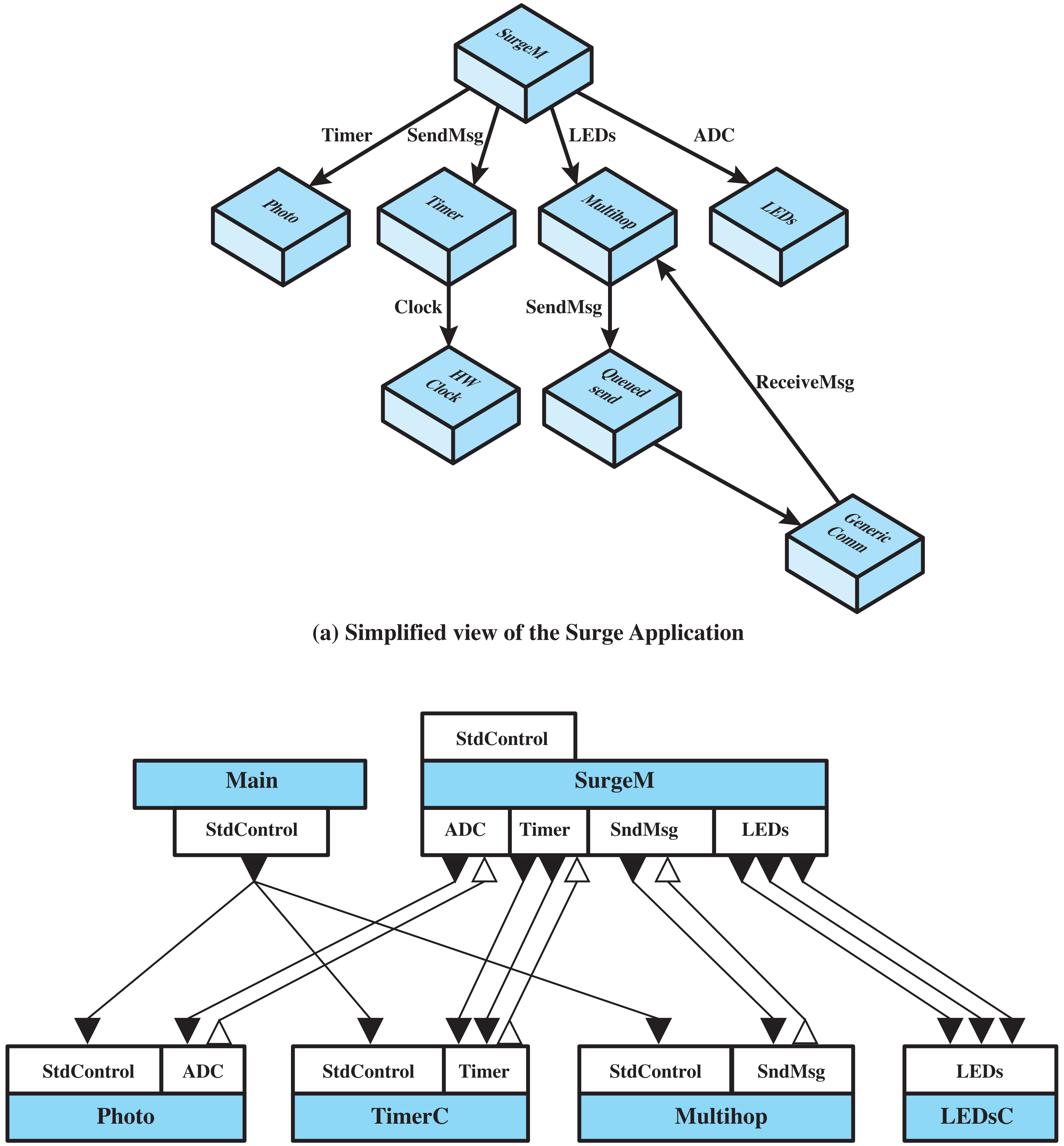
```
module TimerM {
   provides {
      interface StdControl;
      interface Timer;
   }
   uses interface Clock as Clk;
} ...
```

(b) TimerC configuration

```
configuration TimerC {
   provides {
      interface StdControl;
      interface Timer;
   }
}

implementation {
   components TimerM, HWClock;
   StdControl = TimerM.StdControl;
   Timer = TimerM.Timer;
   TimerM.Clk -> HWClock.Clock;
}
```

Figure 13.9 Example Component and Configuration



(a) Simplified view of the Surge Application

(b) Top-level Surge Configuration

LED = light-emitting diode
ADC = analog-to-digital converter

Figure 13.10 Example TinyOS Application

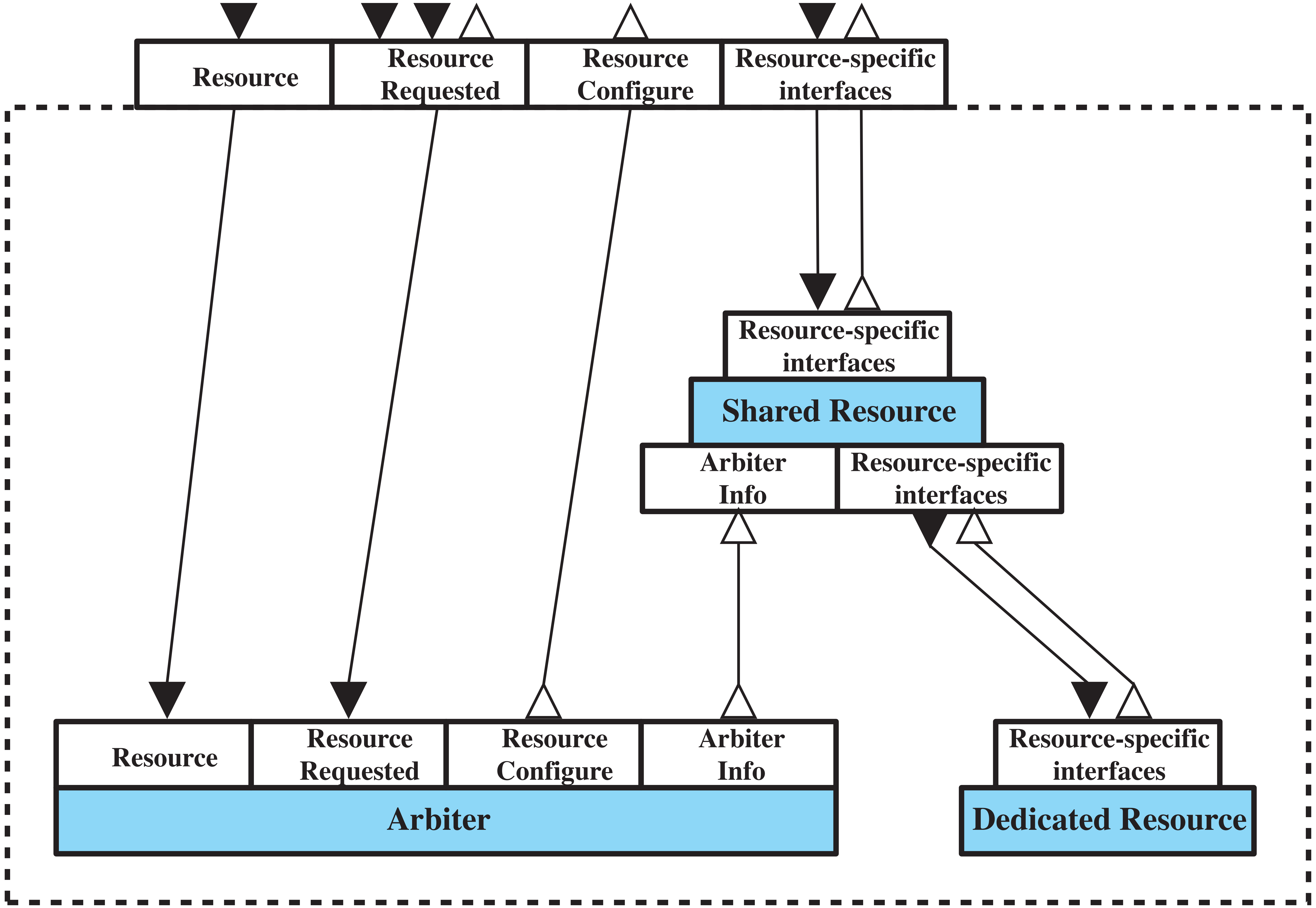


Figure 13.11 Shared Resource Configuration

```
unsigned char buffer_empty = true;
cyg_mutex_t mut_cond_var;
cyg_cond-t cond_var;

void thread_a( cyg_addrword_t index )
{
   while ( 1 ) {    // run this thread forever
      // acquire data into the buffer ...

      // there is data in the buffer now
      buffer_empty = false;

      cyg_mutex_lock( &mut_cond_var );

      cyg_cond_signal( &cond_var );

      cyg_mutex_unlock( &mut_cond_var );
   }
}

void thread_b( cyg_addrword_t index )
{
   while ( 1 ) {    // run this thread forever
      cyg_mutex_lock( &mut_cond_var );

      while ( buffer_empty == true ) cyg_cond_wait( &cond_var );

      // get the buffer data ...

      // set flag to indicate the data in the buffer has been processed
      buffer_empty = true;

      cyg_mutex_unlock( &mut_cond_var );

      // process the data in the buffer
   }
{
```

Figure 13.12 Condition Variable Example Code