S.M. ShamsheerDaula, Dr.K.ESreenivasa Murthy, G Amjad Khan / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 1, Jan-Feb 2012, pp. 887-890

# A Survey on Performance Improvement of Embedded systems designed using ARM family processors

**S.M. ShamsheerDaula** Asst.Professor
**Dr.K.ESreenivasa Murthy** Principal,
**G Amjad Khan** Asst.Professor

G.Pulla Reddy EngineeringCollegeSKTRMC,Kondair.
G.Pulla Reddy EngineeringCollege Kurnool,A.P India.
Andhra Pradesh,IndiaKurnool,AP,India

**Abstract:**

Sticking on with the basic idea, any processing device has its own specialty of virtue. It's the usage that makes the difference depending on the requirement in the application. The general purpose attribute of a Microcontroller is very significant, and shouldn't be overlooked. A general purpose Microcontroller is a very powerful tool that allows a designer to create a special purpose design. The design becomes partially hardware and partially software. This paper has a thing to propose about the running trend of most mobile embedded devices and sensor based applications rely on ARM Cortex processors, where its standards and performance throughputs are given in comparison with other products designed as parallel trend to ARM processors. It also focuses the way in which attention to interdependent detail in intelligent voltage management, cache controller design, conditionally executable instructions, and interrupt control hardware contribute to the ARM architecture's high performance and low power operation.

**General Terms:**32 bit Microcontroller, VCC, PGA, Operating frequencies**.**
**Keywords:**ARM7TDMI, Intel 8051, AT89C51, 8095 microcontrollers, PIC 16C61 Tabulated features.

## 1.Introduction:

A microcontroller can be considered a self-contained system with a processor, memory and peripherals and can be used as an embedded system[1]. The majority of microcontrollers in use today are embedded in other machinery, such as automobiles, telephones, appliances, and peripherals for computer systems. Microcontrollers are used in automatically controlled products and devices, such as automobile engine control systems, implantable medical devices, remote controls, office machines, appliances, power tools, and toys. By reducing thesize and cost compared to a design that uses a separate microprocessor, memory, and input/output devices, microcontrollers make it economical to digitally control even more devices and processes.

Mixed signal microcontrollers are common, integrating analog components needed to control non-digital electronic systems.

At most, the design and architecture of a microcontroller goes with few commonarrangements, like a CPU, XTAL clock, Timers,I/O ports, Serial ports, Reduced memory set.

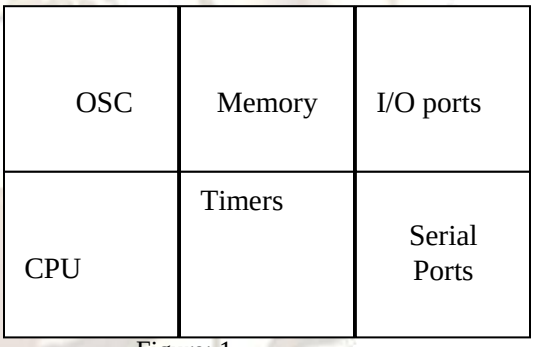


Figure: 1
**Block Diagram of a Microcontroller**

An early 4-bit microcontroller as well the running trend 32-bit microcontroller[1,2] run with relative features as posed in the figure1. The stage set for the microcontrollers most commercial usage was in the reign lead by 8-bit microcontrollers.

## 2. Steps in improving standards:

The range of applications of microcontrollers is viewed from the point of data processing capabilities as follows.

The basic size of data that a microcontroller can handle is 4 bits. Because 4 bits handle the statuses of 0 to 15, the microcontroller mainly handles numerals. As the application field, therefore, a cache register that mainly calculates

decimal numbers can be cited. Because the statuses of 0 to 255 can be handled by 8 bits, an 8-bit microcontroller can use characters such as alphabetic characters, in addition to numerals. Because characters can be used, theapplication of the microcontroller covers English word processors and personal computers.
If 16-bit data can be handled, two-byte characters can also be used. However, a 16-bitmicrocontroller[7,8] is not sufficient for handling data such as sounds andimages. A 32-bit microcontroller can handle sounds and images because much larger data can be handled.

Because physical quantities in the natural world are analog variables, the output signals of a sensor are analog signals. An A/D converter is necessary to use these signals. Some of the output signals of a microcontroller for control applications are 1-bit to perform an ON/OFF control action such as to illuminate or extinguish a light, some are analog signals that control brightness, issue sounds, or control DC motor, while the others are signals that drive an actuator such as a motor or, in some cases, a display unit such as an LCD or fluorescent indicator panel. Microcontrollers are therefore required to have output functions for supporting

To use a microcontroller in a mobile system, it is necessary that the system be organized to be compact. In addition, low-voltage operations and low power consumption are important factors. Microcontrollers for control or embedded applications therefore have functions suitable for various purposes, by providing particular features.

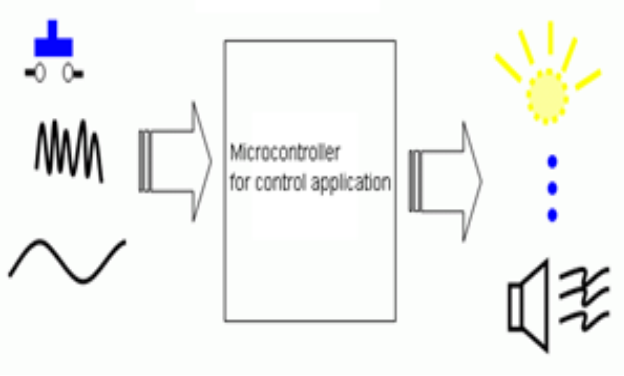


Figure 2: **Operations of Microcontrollers for control applications.**

## 3. Need of ARM Controllers:

ARM's success to date has been largely due to its remarkable performance/power (MIPS/Watt) rating[1], and this will likely continue to be its most critical benchmark for future applications. ARM processors have some characteristics that made them suitable for low power applications, such as: Intelligent Voltage Management, cache micro-architecture. The ARM processor business models are licensed to customers, who integrate them into System-on-Chip (SoC) devices.[9,15,3,4] The most prominent features that make ARM differ from other microcontrollers are, Intelligent Voltage management, Interrupt System, Cache Controller Design and of course its Instruction set.

### 3.1 ARM'S Instruction Set

The RISC (Reduced Instruction Set Computer) design approach had a major influence on the design of the ARM processor [10]. ARM is a RISC machine, also called "Load/Store" type architecture. All ARM instructions are 32 bits long and most of them execute in one clock cycle. However, in order to increase code density in order to compete with other architectures, ARM processors have incorporated a novel mechanism called Thumb (later Thumb2) architecture which provides a 16-bit instruction set that is a compressed form of the original 32-bit ARM instruction set. It is beyond the scope of this discussion to analyze, in detail, the ARM instruction set, but it is possible to just flag several differences from a traditional CPU implementation, that contribute to the increase of code density and faster execution[11,12,13,16].
Although not immediately obvious as a contributor to higher performance and lower power consumption, one unusual feature of the ARM architecture is that most of the instructions can be made to execute conditionally by post-fixing them with the appropriate condition code. Compared to other instruction sets this increases code performance by reducing the number of forward branches. Code performance is measured in clocks per iteration. Less average clocks per iteration reduces power consumption.
The ARM has 16 user-accessible general-purpose registers called r0 to r15 and a current program status register, CPSR. Register r15 contains the program counter, and register r14 is used to save subroutine return addresses (the link register).
The ARM has more than one program status register. In normal operation the CPSR contains the current values of the condition code bits (N, Z, C, and V) and 8 system status bits. When an interrupt occurs, the ARM saves the pre-exception value of the CPSR in a stored program status register (there's one for each of the ARM's five interrupt modes). The ARM runs in its user mode except when it switches to one of its other five operating modes. Interrupts and exceptions switch in new r13 and r14 registers (the so-called fast interrupt switches). When a mode switch occurs, registers r0 to r12 are unmodified.[4,7,14] One of the ARM's most

interesting features is that each instruction is conditionally executed. This ability makes it easy to write compact code.
In the following example, with reference to figure 2 the two SUB instructions are executed only if the conditions GT and EQ are true:
CMP r0, r1 ;compare r0 with r1 and set flags
SUBGT r2, r2, #1 ;if> then r2=r2-1 flags remain unchanged
SUBEQ r3, r3, #1 ;if= then r3=r3-1 flags remain unchanged

**3.2 Model Arrangement of ARM Code**
A small "C" routine and the corresponding ARM Instructions that illustrate the effect of conditional instructions: C source code ARM instructions[1,6,7] unconditional conditional 5 instructions 3 instructions Reducing code by using ARM conditionally instruction execution.

Another specific of the ARM architecture instruction set[5,6] is that data processing instructions do not affect the condition flags of the Program Status Word. However, this can be achieved by post-fixing the instructions with an added "S" after the instruction mnemonic. Here is an example:

```
#include <LPC21xx.H>
#include "Serial.h"
#define CR     0x0D

intsendchar (intch)  {
  if (ch == '\n')  {
    while (!(U1LSR & 0x20));
    U1THR = CR  }
  while (!(U1LSR & 0x20));
  return (U1THR = ch);
}
int uart0_getkey (void)  {
  while (!(U0LSR & 0x01));

  return (U0RBR);
}
```

The obtained information from the sensor
terminals, is further brought up to the possible conversions
into digital format, required by the processor. As an inbuilt provision of theses tools in the ARM structure itself makes a
sense of obtaining peculiar task to its measuring range, the suitable representation of the code is given below as

```
#include <stdio.h>
#include <LPC214x.H>
#include "Serial.h"


  #define ADC_DONE      0x80000000

int temp,temp1;

intread_adc()
        {
                unsigned int i;

          AD0CR = 0x00200308;
                AD0CR |= 0x01000000;
                do
                {
                        i = AD0DR3;

                } while((i & ADC_DONE) == 0);
//ADC_DONE = 0x80000000
                AD0CR &=0xf8ffffff;   //stop ADC NOW
                temp  = (i >>6) & 0x3FF;
                return temp;
}


void delay(int count)
{
int j=0,i=0;

  for(j=0;j<count;j++)
  {
    /* At 60Mhz, the below loop introduces
    delay of 10 us */
    for(i=0;i<35;i++);
  }
}

/*****************/
/* main program */
/*****************/
int main (void)  {       /* execution starts here  */
  char tmp[10];
int i=0;

    uart0_init();
// Initialize UART0
    PINSEL1 = 0x10000000;  // Enabling ADC0.3 pin
    uart0_puts("ADC INTERFACING \r\n");
  while (1) {
     temp=read_adc();                    /* An embedded program does not stop */
```

```
    i=0;
     uart0_puts("\r\nThe ADC Value :");        //
Transffer data to PC through Serial
                                        if(temp==0)
                    tmp[i++]=48;
               else{
     while(temp!=0)
                    {
                    temp1=temp%10;
                    temp=temp/10;
                    tmp[i++]=temp1+48;
                 }
               }
                    i--;
                    while(i>=0)
                    {
                    uart0_putc(tmp[i]);
                    i--        ;
                    }

  delay(100000);
   }
}
```

## 4. Performance Analysis

The shown table is tabulated with
Experimental results on sensor reading operation and comparison with the given controller's architecture features.

| Processor Family | Data Bits | Clock Rate | Memory Organisation | MIPS/ MHZ | Power Consumption |
|---|---|---|---|---|---|
| Intel 8095 | 16 | 5 MHz | Harvard | 0.3 | 5.25 V |
| PIC1XFXX | 16 | 4 MHz | Harvard | 0.42 | 4.75 V |
| Tri Core XX | 16/32 | 33 MHz | Harvard | 0.8 | 4.75 V |
| ARM7TDMI-LPC21XX | 16/32 | 66 MHz | Vonneuman | 0.9 | 2.5 V |
| ARM9TDMI | 32 | 200 MHZ | Hravard | 1.15 | 2.5 V |

Table:1**Comparisons of figured Controller features.**

## 5. Summary and Future works:

The running trend of most mobile embedded devices and sensor based applications rely on ARM Cortex processors, where its standards and performance throughputs are given in comparison with other products designed as parallel trend to ARM processors.Moreover its small size and low power consumption fulfill the requirements.After constantly
theoretical analysis and simulations, the result shows that the
ARM controllers can well meet the expected demands.
Related to the above simulation work
Further the ARM can excel in parallel calculations for the image processing.

## 6. References

[1]. J. G. Li, X. Y. Nie, and Z. M. Jiang, "Development Specifications for ARM Application Systems", Beijing: the Tsinghua University Press, 2005.
[2] Michael Barr, "Programming Embedded Systems in C and C++" , America: O'Reilly, 2007.
[3]. http://www.powercube.com.
[4]. Heath, Steve (2003). *Embedded systems design*. EDN series for design engineers (2 ed.). Newnes. pp. 11–12.ISBN 9780750655460.
[5] http://books.google.com/books?id=BjNZXwH7HlkC.
[6]. http://www.seattlerobotics.org/encoder/sep07/basics.html.
[7] Steve Furber, Stephen B., ARM system-on-chip architecture, Addison- Wesley, 2008.
[8]. Intel® PXA255 Processor Developer's Manual, http://www.intel.com/design/pca /applicationsprocessors/manuals/278693.htm.
[9] Samsung S3C2410x User Manual, http://www.samsung.com/Products/Semiconductor/SystemLSI/Mobile tions/ MobileASSP/MobileComputing /S3C2410X/um_s3c2410s_rev12_030428.pdf .
[10] Cirrus Logic Maverick EP7312 Manual, http://www.cirrus.com/en /pubs/proDatasheet /EP7312-5.pdf
[11].I2C bus specification, http://www.semiconductors .phiips.com/acrobat/literature/9398/39340011.pdf.
[12] LIU Hong-li, "The Research and Experiment of the Embedded SystemμC/OS-IIonPC,"Journal of Shanghai University of Electric Power,Magn. China,vol.5(7),pp.275-248,June 2009.
[13]. ZHANG Shi, DONG Jianwei, SHE Lihuang, "Design and developmentof ECG monitor'ssoftwaresystem,"Computer Engineering,Magn.China,vo33(9), pp.277-279, May2007.
[14] Todd Austin, Trevor Mudge, Dirk Grunwald. "An ARMloadofSimpleScalar/ ARM
[15] L. G. Zhou, . Foundation and Practice for ARM Microprocessors, Beijing: Beijing University of Aeronautics and Astronautics Press, 2008
[16] J. G. Li, X. Y. Nie, and Z. M. Jiang, "Development specifications for ARM Application Systems", Beijing:Tsinghua University Press, 2009