Abstract. Today, many students are study computer science. Although they study different courses, they rarely do homework or projects related to real world problems. In this paper, we design a project to show some of the problems which are occured during implementing a real word project to the students.

In this project, the students are asked to implement the FFT (Fast Fourier Transform) algorithm on a personal computer and then port the program to a smartphone using J2ME (Java 2 Micro Edition) programming language. By doing this project, the undergraduate computer science students will become familiar with the limitations of small devices such as smartphones. They also learn the process of porting a program from a personal computer to a smartphone.

Keywords. Computer Science Students, DFT (Discrete Fourier Transform), FFT (Fast Fourier Transform), J2ME (Java 2 Micro Edition), Smartphones.

1. Introduction

One of the important inventions of the 20th century is Computer. The Computer changed human life in less than 50 years. Today, computers are used in nearly all places. While most people think that only laptops, workstations and servers are computer, computers are widely used as embedded systems. For example, there is a small computer in a microwave oven or washing machine. Embedded systems market has an important share of computer related markets.

Embedded systems have architecture similar to general purpose computers: they have CPU and memory and are programmed to do a special task with programming languages such as C++ and Java. In embedded systems, there are a number of limitations, such as limited computing power, limited RAM and limited instruction set. For example, most of laptops and workstations which are produced now have 1 GB or more of high speed (for example DDR2) RAM, but embedded devices have a very limited RAM, usually less than 1 MB. Embedded processors usually don’t support floating point arithmetic.

A device which is something between general purpose computer and usual embedded system is smartphone. At first, mobile phones were merely a device for communication and talking with each other. But now, mobile phones’ features have increased and smartphones are used widely. Because of the advances in mobile phone field, especially after the appearance of smartphones, the mobile software market formed and software manufactures started developing programs for mobiles and the new capabilities of mobile phones enjoyed a lucrative market. Mobile programs can be designed in various fields such as games, play and edit films, photos, and music, organizers and location finders.

Smartphones have powerful processors (for example 300 MHz ARM processors) and relatively big memory (for example 128 MB of RAM) in comparison to embedded systems, but they are very limited in comparison to general purpose computers which used by students.

During academic study, students usually work with general purpose computers; learn how they work and how to write programs for them. Because of the embedded device limitations, programs and algorithms which work well on students computers, may have poor performance or even do not work at all on embedded devices. But writing programs for an embedded device like a washing machine microcontroller is not easy and students cannot try writing programs for them easily.
Considering the similarities between embedded systems and smartphones, students can use smartphones as an example of embedded systems and become familiar with their limitations. As mentioned earlier, today the mobile phones software is a big market, so there are many tools and IDE (Integrated Development Environment) which help programmers to write programs for mobile phones. Moreover, well known languages such as C++ and Java are supported well and compilers for them are available, and most of the mobile phones which are produced support Java.

But small devices such as mobile phones cannot support all of the features which are available in a normal programming language such as C++ or Java. To solve this problem, a smaller version of these languages is designed which can be supported by small devices. An example is J2ME (Java 2 Micro Edition) is a smaller version of Java language which is designed for small devices such as mobile phones, PDAs and embedded devices.

J2ME is based on Java and is a subset of that, so anyone who is familiar with Java can write program for J2ME. In addition, programs which are written for J2ME can be easily compiled and emulated on normal computers, without any special hardware. This is one of the reasons that made J2ME the de facto standard for mobile phone games. So students, which usually learn Java in computer programming courses, can write programs for J2ME without needing to learn a new language. They only must learn the features which are not available in J2ME. On the other hand, most of the new mobile phones support J2ME, so students can test the programs which they wrote on their personal mobile phones.

Therefore students can practice writing programs for embedding devices by using J2ME language and run them on mobile phones. They can see embedded device limitations such as limited available memory and low computation power.

The goal of this project is to give an opportunity to students to write a program for an embedded device. We choose the J2ME language, because it is the de facto standard in mobile game market and students can easily write programs for that because they know Java language. In addition, there are freely available IDE, SDK (Software Development Kit) and compilers for J2ME from different mobile vendors such as Nokia and Sony-Ericsson.

We select smartphones as target embedded devices, because they are not very limited and can be easily used for testing programs. In addition, most of the mobile phones which students have support J2ME and they can also test programs on their own mobile phones, which made the experiment more enjoyable.

An important part in designing this project is choosing a program to be written by students. In the programs which students write as final projects of computer programming courses, they usually spent a lot time for designing user interface. Because the user interface design need a lot of time, we decided to choose an algorithm with high calculations to be implemented. Among the different algorithms, we select the DFT (Discrete Fourier Transform).

The main reason for selecting DFT is that DFT is widely used in computer science, computer engineering and electrical engineering and it is a good experience for students to implement one of the fundamental algorithms which they will use in future. There is no need to be familiar with Fourier transform to do this project. We will tell the students how the DFT is calculated and they implement that. They will be learning the fundamentals of Fourier transforms in courses such as “Signals and Systems” and its applications in courses such as “Image Processing”.

Another reason for selecting DFT is to see the differences between different algorithms which do identical task. First they will implement DFT by the formula which defines that. Then they will use FFT (Fast Fourier Transform) algorithm to calculate DFT.

When they implement both methods, they can measure the time which used to done the task and their differences. By porting the algorithm to J2ME, they can see the differences between running an algorithm on desktop computer and mobile phone. They can also see the limitations of embedded devices by trying to calculate the DFT of a big signal.

In continue, at first we will introduce the DFT. After that, we explain the FFT algorithm and represent the recursive pseudo-code of that. Then we express the steps which student must be done to do the project in third section. At the end, we will have a conclusion.
2. DFT and FFT

In this section, at first we define DFT, then we will explain FFT algorithm and finally represent the recursive pseudo-code of that. A more complete tutorial for Fourier Transform, Discrete Fourier Transform (DFT) and Fast Fourier Transform (FFT) are available on Wikipedia [4-6]. Comprehensive study of Fourier transform is available in most of Signal and Systems books, such as [3]. Chapter 30 of [1] is about DFT and FFT and how the FFT algorithm is derived from DFT formula and how to implement FFT efficiently. The pseudo-codes in this section are mostly derived from [1].

2.1. DFT (Discrete Fourier Transform)

Fourier Transform is a linear transform which decompose a function in terms of a sum of sinusoidal functions. The decomposition of a function to sum of sinusoidal functions is usually called Fourier analysis. Fourier synthesis is the reverse of Fourier analysis and constructs a function from a sum of sinusoidal functions. The name of this transform comes from “Joseph Fourier” who introduced the Fourier series.

Fourier analysis has many applications in physics, mathematics, electronics and computers. In computer and electrical engineering, it used in different applications such as signal processing and image processing.

The Fourier transform can be continuous or discrete, periodic or aperiodic. So there are four type of Fourier transform. Table 1 shows these transforms names and properties of function in Frequency domain.

Because computers are digital devices, Discrete-time Fourier Transform or simply Discrete Fourier Transform (DFT) is more important in computer science and engineering. As it is shown in Table 1, DFT of discrete aperiodic signals is continuous in frequency domain. So it is not suitable for use in computers.

Aperiodic signals such as x[n] are usually zero outside of an interval 0 ≤ n ≤ N. So we can assume that they are repeated outside that region. If we call the DFT of original aperiodic signal with X(ω) and DFT of signal created by repeating the signal outside that interval with \( \tilde{X}[k] \), it can be proved that \( \tilde{X}[k] = \frac{1}{N} X(e^{j(2\pi k / N)}) \). So, instead of calculating the DFT of an aperiodic signal, which is a continuous function, we calculate the DFT of the signal created by repeating the signal outside of the interval which is non-zero and call this as DFT of that signal.

The DFT of a signal x[n] which is non-zero for 0 ≤ n ≤ N and zero outside of this interval, is defined as follow:

\[
X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi \frac{k n}{N}} \tag{1}
\]

The quantity \( e^{2\pi j / N} \) is called the n th root of unity and its symbol is \( \omega_N \). The reason that this value named n th root of unity is because we have \( \omega_N^N = 1 \). So the DFT can be written as:

\[
X[k] = \sum_{n=0}^{N-1} x[n] \omega_N^{kn} \tag{2}
\]

Because the X[k] is a periodic function, it is sufficient to save X[k] only for 0 ≤ k ≤ N. Therefore the DFT of a vector with length n is a vector with length n.

In the calculation of DFT, we need to calculate the complex power of numbers. The exponential of a complex number defines as:

\[ e^{j\theta} = \cos(\theta) + j\sin(\theta) \tag{3} \]

<table>
<thead>
<tr>
<th>Name</th>
<th>Time Domain</th>
<th>Frequency Domain</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Continuous) Fourier transform</td>
<td>Continuous, Aperiodic</td>
<td>Continuous, Aperiodic</td>
<td>( X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft} dt )</td>
</tr>
<tr>
<td>Fourier series</td>
<td>Continuous, Periodic</td>
<td>Discrete, Aperiodic</td>
<td>( c_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} x(t) e^{jnt} dt )</td>
</tr>
<tr>
<td>Discrete-time Fourier transform</td>
<td>Discrete, Aperiodic</td>
<td>Continuous, Periodic</td>
<td>( X(\omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n} )</td>
</tr>
<tr>
<td>Discrete Fourier transform</td>
<td>Discrete, Periodic</td>
<td>Discrete, Periodic</td>
<td>( x[k] = \sum_{n=0}^{N-1} x[n] e^{j2\pi \frac{k n}{N}} )</td>
</tr>
</tbody>
</table>
2.2. FFT (Fast Fourier Transform)

Calculating the DFT using equation (2), with a trivial method, need \( O(n^2) \) operations. Because the DFT has many applications, we need fast algorithms to calculate it. One of the algorithms which is widely used to calculate DFT is FFT that is acronym for Fast Fourier Transform. FFT will compute the DFT of a signal with \( O(n \log n) \) operations. The FFT algorithm is a divide and conquer algorithm. If we assume that \( N \) is even and create two signal \( x_0[n] \) and \( x_1[n] \) so that \( x_0[n] \) contains even-index samples of \( x[n] \) and \( x_1[n] \) contains odd-index samples of \( x[n] \), and \( X_0[k] \) and \( X_1[k] \) are the DFT of \( x_0[n] \) and \( x_1[n] \), then we have:

\[
X[k] = \frac{1}{2} X_0[k] + \frac{1}{2} \omega^n X_1[k]
\]  

(4)

Equation (4) tell us that we can compute the DFT of odd-index samples and even-index samples separately and then combine the results to produce the DFT of original signal. The following pseudo code shows the recursive method for computing FFT.

**Recursive-FFT(x):**

1. \( n = \text{length}(x) \)
2. if \( (n=1) \) then
3. return \( x \)
4. \( \omega_N = e^{2 \pi i / N} \)
5. \( \omega = 1 \)
6. \( x_0 = (x[0], x[2], \ldots, x[n-2]) \)
7. \( x_1 = (x[1], x[3], \ldots, x[n-1]) \)
8. \( y_0 = \text{Recursive-FFT}(x_0) \)
9. \( y_1 = \text{Recursive-FFT}(x_1) \)
10. for \( k = 0 \) to \( n/2 - 1 \) do
11. \( y[k] = \frac{1}{2} y_0[k] + \frac{1}{2} \omega^{k} y_1[k] \)
12. \( y[k + n/2] = \frac{1}{2} y_0[k] - \frac{1}{2} \omega^{k} y_1[k] \)
13. \( \omega = \omega \cdot \omega_N \)
14. return \( y \)

In calculating the FFT of signal from the FFT of two sub-signals, we assume that \( N \) is even. If we want to repeat this step until we reach a signal with one sample only (which the DFT of signal is the value of that sample), the \( N \) must be a power of 2. In cases which \( N \) is not a power of 2, we can add enough zeros to signal so that \( N \) becomes a power of 2.

3. Project description

As mentioned in the introduction, the goal of this project is to show embedded systems limitations. In addition, this project will enable the students to write a program for an embedded device. The students will implement the DFT in Java language.

First they implement the DFT using equation (2) on a desktop computer in Java. They will use J2SE (Java 2 Standard Edition) in this part. J2SE is the version of Java for writing program on normal desktop computers. In this part, they will need some programming skills. For example, they must implement the complex multiplication and power. After that, they will implement the FFT algorithms. Now they will use both methods to calculate some sample signals and compare the time required to calculate the Fourier transform of each signal. For sample signals, we can use simple signals such as a sine function, a constant function and a real sound signal. The test signals must have different lengths, from very short (for example 10 sample), to very long (for example 1 minute of audio).

After testing the algorithms on a desktop computer, they must port their programs to J2ME. Java 2 Micro Edition (J2ME) is a version of Java language for small devices. In this process, they will learn which features are not available in embedded systems. The target machine which their programs must be run on it is a mobile phone. There are many choices for mobile phone, but we suggest to select a popular phone with price less than 200$, because some of expensive mobile phones have powerful hardware.

Among different brands, we prefer Nokia because we found more tools and information regarding Java programming on Nokia phones. A good place to find these tools is Nokia Forum [2].

When they ported the programs to mobile phones, they will compute the Fourier transform of the same signals which they use in first part. Now they can compare the time required to compute the DFT in both machines with each algorithm. They will see the differences between algorithms which do a same task and also the differences between processing powers of embedded systems and desktop computers. In addition, they will be faced some problems in calculating DFT (or FFT) of long signals in mobile phone, because mobile phones have a limited memory.
They can also compute the Fourier transform of the signals in MATLAB and compare the results with their programs. This comparison can show them the round-off errors which are introduced in their calculation and usually are not visible in design times.

4. Conclusion:

This project is preferred for Computer Science students, because this project improves both theoretical and experimental aspects of students’ abilities.

In this project, the students should implement the Discrete Fourier Transform (DFT) and Fast Fourier Transform (FFT) on the both PC and mobile phone and compare the time required to run FFT on each platform. The programs must be written in Java and then ported to J2ME for running on mobile phone. After porting the program to J2ME, they can run the program on the mobile phone and measure the time required to calculate the FFT. Because most of available mobile phones support J2ME, the student can also run the program on their own cell phones, in addition to lab’s mobile phone!

By doing this project, students learn more about Fourier transform and how it is implemented. They also become familiar with the differences between PCs and embedded devices like mobile phones, for example the limited memory of embedded devices and their limited computation power. By comparison of DFT and FFT results, they will see that how choosing a good algorithm can improve overall performance of a system. Also they will learn that algorithms and programs which work well on a desktop computer, may be unusable in embedded devices.

This project can also make students familiar with mobile phones programing and prepare them to write programs for them. With current growth of mobile phones market and especially mobile phones software market, there will be more demand for mobile phone programmers and projects similar to this one can attract more students to this field.

The cost of devices which are this project needs is low, because each group needs a mobile phones, which is about 200$, a computer, which is usually available in all labs, Java software development environment which free one can be used such as Eclipse, and finally compilers for both desktop computer and mobile phones which for both, there are free Java compilers available.

This project can be done in three lab session. In first session, the theoretical background which are required such as Fourier transform equations is thought and a number of references are offered to them for more reading. In second session, they will implement both DFT and FFT in Java and run them on desktop computers. The third session is for porting the programs to mobile phones. To prepare students for third session, some references which describe J2ME and its differences with normal Java is given to students and they can read them to become familiar with J2ME.

5. References
