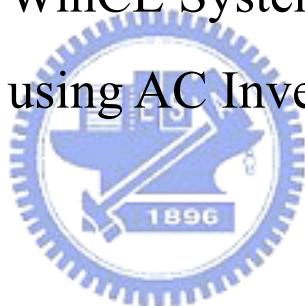


國立交通大學

機械工程研究所
碩士論文

WinCE 嵌入式系統於變頻式電動跑步機之開發
Embedded WinCE System for Treadmill
using AC Inverter



研究生：余劭軒

指導教授：成維華 教授

中華民國九十四年七月

WinCE 嵌入式系統於變頻式電動跑步機之開發

Embedded WinCE System for Treadmill using AC Inverter

研究生：余劭軒

Student： Shao-Hsuan Yu

指導教授：成維華 博士

Advisor： Dr. Wei-Hua Chieng

國立交通大學



Submitted to Department of Mechanical Engineering
College of Engineering
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master of Science
in

Mechanical Engineering

July 2005

Hsinchu, Taiwan, Republic of China

中華民國九十四年七月

WinCE 嵌入式系統於變頻式電動跑步機之開發

學生:余劭軒

指導教授:成維華

國立交通大學機械工程學系

摘要

電動跑步機為目前很受歡迎的健身器材,本論文之主要目的在於使用嵌入式系統做為電控系統,以WinCE.NET作為嵌入式作業系統,由C#開發使用者介面以及整合多媒體的使用,網路傳輸功能藉以改善傳統跑步機以單晶片 8051 為電控系統,由於記憶體容量限制,不易與網路和多媒體整合之缺點.而在動力系統方面,由於變頻器的改善,以交流感應馬達配合向量控制的技術,藉以得到較佳之輸出性能.

Embedded WinCE System for Treadmill using AC Inverter

Student: Shao-Hsuan Yu

Advisor: Dr. Wei-Hua Chieng

Institute of Mechanical Engineering
Natuonal Chiao Tung University

Abstract

The electric treadmill is the popular apparatus at the present day. The main purpose of the thesis is to use the embedded system as the electric control system and develops the application of the user interface and integrates multimedia, internet transmission by the programming language C#. By doing so, we can improve the traditional treadmill's disadvantage such as difficult to integrate multimedia and internet due to the leak of memory when using 8051 as the electric control system. In the aspect of power resource, due to the improvement of inverter, we use the AC (Alternative Current) induction motor with the technique of vector control to get the better output performance.

Acknowledgement

I would like to express my deep sense of appreciation to my advisor, Dr. Wei-Hua Chieng, for his outstanding guidance and earnest inspiration during my graduate study and research.

The concern and support from my family and friends is of great significance to me. I dedicate my sincere gratitude to every one of them.



Contents

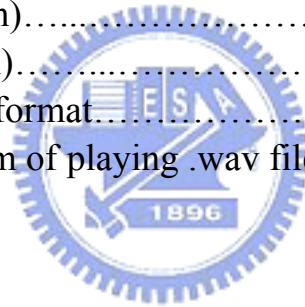
摘要.....	i
Abstract	ii
Acknowledgements	iii
Contents	iv
List of Figures	vi
List of Tables	vii
Chapter 1 Introduction	1
1.1 Motivation.....	1
1.2 The classification of treadmill.....	2
1.3 The treadmill nowadays.....	2
1.4 Improvement.....	4
Chapter 2 Hardware Architecture	5
2.1 Development board.....	5
2.1.1 CISC VS RISC.....	5
2.1.2 Common Microprocessors.....	7
2.1.3 Trade off.....	9
2.2 Three phase induction motor.....	10
2.2.1 The control theorem of inverter.....	11
2.2.2 V/F control theorem.....	11
2.2.3 Vector control theorem.....	13
2.2.4 The derivation of vector control.....	13
2.2.5 The comparison between V/f and vector control.....	16
2.3 The electric and mechanical system.....	17
Chapter 3 Software Architecture	18
3.1 Embedded System.....	18
3.2 Embedded Operating System.....	18
3.3 WinCE.NET.....	21
3.4 Development Tools.....	23
3.4.1 Win32 API.....	24
3.4.2 .NET Compact Framework.....	26
3.5 Platform Invoke (P/Invoke).....	29
3.5.1 Marshaling Data.....	30
3.6 ActiveSync.....	30
3.7 Platform Builder.....	31
3.8 The procedure of developing application.....	31
Chapter 4 Implementation	33

4.1 Architecture.....	33
4.2 Communication between PCM-9575 to Inverter EI-8001.....	33
4.2.1 Terminal control.....	34
4.2.2 Serial communication control.....	34
4.2.3 Acceleration time, Deceleration time.....	36
4.2.4 Terminal control v.s. serial communication control.....	36
4.2.5 The process of start and stop.....	37
4.3 The function of the treadmill.....	38
4.3.1 Mode of the treadmill.....	38
4.3.2 The importance of the Warm Up and Cool Down.....	43
4.4 Http Communication.....	43
4.5 Play the wave file format.....	44
4.5.1 The Wave File Format.....	45
4.6 GDI.....	46
Chapter 5 Conclusion.....	47
Reference.....	48



List of Figures

Figure 2.1 PCM-9575.....	50
Figure 2.2 The equivalent circuit of single phase induction motor.....	50
Figure 2.3 Mechanism of the treadmill.....	51
Figure 3.1 Direct Connection.....	51
Figure 3.2 Connect type.....	52
Figure 3.3 Picture of ActiveSync.....	52
Figure 4.1 The whole architecture of the treadmill.....	53
Figure 4.2 The program flow of the treadmill.....	54
Figure 4.3 Main (function).....	55
Figure 4.4 Data Setting (function).....	55
Figure 4.5 Music Mode (function).....	56
Figure 4.6 Basic Mode (function).....	56
Figure 4.7 Set Time (function).....	57
Figure 4.8 Walking Course (function).....	57
Figure 4.9 Cardio (function).....	58
Figure 4.10 Race (function).....	58
Figure 4.11 The wave file format.....	59
Figure 4.12 The mechanism of playing .wav file.....	59



List of Tables

Table 2.1 The comparison of the development board.....	60
Table 2.2 The specification of PCM-9575.....	61
Table 3.1 Comparison of the development tool.....	61
Table 4.1 The important register addresses of Inverter EI-8001.....	62
Table 4.2 The line command register of FA00.....	63
Table 4.3 Walking course.....	63
Table 4.4 Interval course.....	64
Table 4.5 Running course.....	64
Table 4.6 Cardio 20min.....	64
Table 4.7 Interval 25min.....	65
Table 4.8 Fat Burn 30 min.....	65
Table 4.9 Endurance 35min.....	66



Chapter 1 Introduction

1.1 Motivation

In recent years, with the improvement of material life, many people in our country are over-nourished. And the weight is increasing, too. Excessive fatness may cause a lot of physiological diseases, such as heart disease, hypertension and diabetes. The best way to avoid diseases is to take exercise properly. Exercise is good to health. It accelerates the circulation of blood and promotes the development of muscles.

There are many forms of exercise. Taking exercise outdoor will be the best, but sometimes it may have some restrictions such as climate, place, etc. So in the modern society, it is the trend to exercise indoor. And it may provide a safety environment to ensure that people can exercise without danger. Taking exercise in gym is more and more popular nowadays.

In numerous exercise equipments, one of the most popular exercise equipments is treadmill; it provides many functions, courses, and programs that the users can choose. And it helps the users to obtain the exercise effect they want easily.

Electric treadmill, from the design of the mechanism, IC (Integrated Circuit) drive to the manufacture of the motor, is a complete application to automatic integration. And the research of this topic is worth doing, too.

1.2 The classification of treadmill

From the viewpoint of power resource, the treadmill can be divided into two kinds: the manual-type and the motorized-type. The manual-type treadmill is powered by the users. There are several kinds of manual-type's treadmills at present, such as flat-bed type and the roller type. The motorized-type treadmill, as implied by the name, is powered by the motor [1][2][3][4], and the DC (Direct Current) brush motor is the mainstream in today's electric treadmill. The advantages of the DC brush motor are that the control driver is easy to implement and the cost of the motor is less than other kinds of motors. But there are still some shortages of this kind of motors: the problem of heat dissipation and sparking and the mechanical contact between the brush and the commutator will cause attrition to the components. The other kind of the motor is AC (Alternative Current) induction motor; compared with the DC brush motor, the AC induction motor is brushless, so this kind of motor does not have the problem of sparking, and it has the advantage of maintenance. In the past, due to the driver of the inverter is not good enough, the performance of the AC induction motor is worse than the DC motor. But with the improvement of the semi-conductor component such as IGBT (Insulated Gate Bipolar Transistor), and the control strategy such as vector control [5], the control ability is close to the DC brush motor. And it will become more and more popular for the treadmill's motor in the future.

1.3 The treadmill nowadays

Before the design of the user interface and function, we should reference the good design of the treadmill in existence first. And the summarized result is shown as follows:

(1) The control panel:

Using the LED to constitute the panel is the trend nowadays.

(2) The motor of the electric treadmill:

The DC brush motor is the mainstream of the electric treadmill.

(3) The important parameters shown in the panel:

- Time: the total time that the user uses the treadmill.
- Distance: the total distances.
- Calories: the consuming calories.
- Speed: the current speed.
- Incline: the current incline.
- Heart Rate: the measured heart rate of the user.

(4) The function of the treadmill:

- Manual:

The frequently used mode of the treadmill.

- Basic Control:

This mode has the function of “Set Time”, “Set Calorie”, and “Set Distance”, and the user can control the total time, distances, and calories they want.

- General Course:

This mode has some courses such as “Walking Course”, ”Running Course”, and “Interval Course”. The course has

the value of incline by default and the value of incline can not be changed by the user. The users can adjust the value of speed only.

- **Advanced Course:**

This mode has some courses such as “Cardio”, “Interval”, and ”Fat Burn”. In this mode, there are default values of speed and incline, and at the beginning several minutes of the program, the value can not be changed by the user; it’s the “Warm Up” section. At the end of the program, there is a cool down section. At the middle of the program, there are default values of speed and incline, and the user can adjust both of the values by themselves.

- **Race Mode:**

The user takes a competition with an opponent.

(5) **Data Setting:**

To input the user’s data such as age, weight, Level, etc...



1.4 Improvement

From the previous section 1.2, we think that the function of the treadmill can be said to be complete, and we can do less in this aspect; we can use the 15 inch touch screen to replace the LED in order to have more friendly user interface, and in the other aspect, we can strengthen the ability of connecting to the internet and the use of multimedia by the use of the embedded system [6] as the control system.

Chapter 2 Hardware Architecture

2.1 Development board

To develop the embedded system, it's inevitable to choose the target board. It must be careful to choose the target board, because it catches up in the development environment and the technical support.

2.1.1 CISC V.S. RISC [7]



In the development of the embedded system microprocessor, it can be divided into two kinds of architecture according to the characteristic of the instruction: CISC (Complex Instruction Set Computer) and RISC (Reduced Instruction Set Computer).

RISC architecture makes use of a small set of simplified instructions in attempt to improve performance. These instructions consist mostly of register-to-register operations. Only load and store instructions access memory. Since almost all instructions make use of register addressing, there are only a few addressing modes in a reduced instruction set computer and there are a large number of general-purpose registers. Another way in which reduced instruction set computers sought to improve performance was to have most instructions complete execution in one machine cycle. Pipelining was a key technique in

achieving this. Pipelining allows the next instruction to enter the execution cycle while the previous instruction is still processing.

The advantages of a reduced instruction set computer:

- Faster.
- Simpler hardware.
- Shorter design cycle due to simpler hardware.
- Lower cost since more parts can be placed on a single chip.

The disadvantages of a reduced instruction set computer include:

- Programmer must pay close attention to instruction scheduling so that the processor does not spend a large amount of time waiting for an instruction to execute.
- Debugging can be difficult due to the instruction scheduling.
- Require very fast memory systems to feed them instructions.

CISC used microcode to simplify the computer's architecture. In a micro programmed system, the ROM contains a group of microcode instructions that correspond with each machine-language instruction. When a machine language instruction arrives at the processor, it executes the corresponding series of microcode instructions. Microcode acts as a transition layer between the instructions and the electronics of the computer. This also improved performance, since instructions could be retrieved up to ten times faster from ROM than from main memory. Other advantages of using microcode included fewer transistors, easier implementation of new chips, and a micro programmed design can be easily modified to handle new instructions sets. Another characteristic of complex instructions set is their variable-length instruction format.

Advantages of CISC include:

- Less expensive due to the use of microcode.
- Upwardly compatible because a new computer would contain a superset of the instructions of the earlier computers.
- Fewer instructions could be used to implement a given task, allowing for more efficient use of memory.
- Simplified compiler, because the microprogram instruction sets could be written to match the constructs of high-level languages.
- More instructions can fit into the cache, since the instructions are not a fixed size.

Disadvantages of CISC:

- Instruction sets and chip hardware became more complex with each generation of computers; since earlier generations of a processor family were contained as a subset in every new version.
- Different instructions take different amount of time to execute due to their variable-length.
- Many instructions are not used frequently; approximately 20% of the available instructions are used in a typical program

2.1.2 Common microprocessors

From the above discussion, we can divide the microprocessors into two kinds: RISC and CISC. The following are the common microprocessors in the market.

- (1) RISC architecture

- ARM

Each product family consists of high-performance, energy efficient designs built to handle the performance demands of today's increasingly complex electronics applications. ARM offers the industry's broadest range of 16/32-bit embedded RISC cores that are grouped into a range of families: the ARM7, the ARM9, the ARM10, and ARM11 of microprocessor cores.

- OMAP

Produced by TI (Texas Instrument), from smart phones to high-end, multimedia-rich 3G wireless handsets and PDAs, the scalable family of OMAP processors delivers the best combination of high-performance and ultra-low power consumption.

- XScale

The Intel XScale microarchitecture is based on a new core which is compliant with ARM-5TE. The microarchitecture surrounds the core with instruction and data memory management units; instruction, data, and mini-data caches; write, fill, pend, and branch target buffers; power management, performance monitoring, debug, and JTAG units; coprocessor interface; 32K caches, MMUs.

- MIPS

MIPS is a RISC microprocessor architecture developed by MIPS Computer Systems Inc. MIPS designs have found broad application in embedded systems, Windows CE devices, and Cisco routers. The Nintendo 64 console, Sony PlayStation console, Sony PlayStation 2 console, and Sony PSP handheld system use MIPS processors. By the late 1990s it was estimated that one in three of all

RISC chips produced were MIPS-based designs.

(2) CISC architecture

- Mainly the x86 instruction, such as AMD-K6E, and Advantech's x86 products.

2.1.3 Trade off

When choosing the development board to the control system of the treadmill, we should consider the following points:

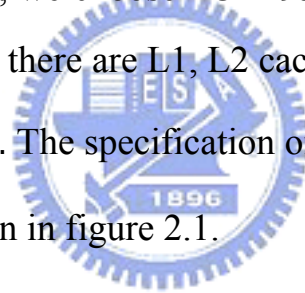
- CPU: The speed of the CPU is an important point that we should notice; we use the programming language C# to develop the application, and it may need the high-speed CPU to prevent the occurrence of latency, and the most important thing of the treadmill is safety, and we should prevent the injury owing to latency. Compared with the CISC architecture's CPU, the fastest CPU of the RISC architecture is XScale, and it only reaches 400 MHz. It's better to choose CISC architecture's CPU to have the high speed CPU.
- Cache: some CISC architecture has larger cache than RISC architecture.
- Heat dissipation: The RISC architecture has the better performance of heat dissipation.
- Display chipset: We try to make the user interface more friendly by use of graphical images, so it's better to have display chipset; some CISC architecture has the

display chipset.

From the above discussion, we need the high speed CPU, and it's better to have cache, and display chipset, so CISC architecture is the better choice, but in the aspect of heat dissipation, the CISC architecture has the worse performance, so CISC architecture usually has the fan to improve the heat dissipation problem.

In our treadmill system, we want no fan in the development board, and there are some x86 based development boards that support fanless corresponding to our requirement. The table 2.1 is the comparison between some Advantech's products. (: good, : acceptable, X: bad)

Under the comparison, we choose PCM-9575 to be the development board; The CPU is 667 MHz, and there are L1, L2 cache, and it can operate without fan under the temperature 60°C. The specification of the PCM-9575 is shown in table 2.2 and the picture is shown in figure 2.1.



2.2 Three phase induction motor

Generally speaking, the induction motor is robust and the cheapest of all electric rotating machines. The stator has a three-phase winding and there are two types of rotor construction: cage-type rotor, and wound-type rotor. Although the construction is simple, its mathematical model has the following characteristic: nonlinear, time-varying, high-order coupling. So the control method is more complex. In the past, due to the inverter is not good enough, the control performance of the AC induction motor is worse than DC motor.

2.2.1 The control theorem of inverter

From the induction motor theorem [5] we know, changing the speed of the magnetic field produced by stator will also change the speed of magnetic field of the rotor, and the rotating speed of the rotor will change, too.

From the equation:

$$\omega_{\text{m}} = \frac{120f_{\text{s}}}{\text{P}} \quad (2-1)$$

ω_{m} : rotating speed of the motor (rpm)

f_{s} : supplying power frequency (Hz)

P: poles of the motor

We can change the rotating speed of the motor by changing the poles of the motor or by changing the supplying power frequency on the stator.

But it's not easy to change the poles after the motor is produced. So we usually change the power frequency we supply to change the rotating speed of the motor. The inverter can change the frequency to achieve the goal of changing the rotating speed. In the present day, there are two control strategies that are using commonly in the inverter: V/f control and the vector control.

2.2.2 V/F Control theorem

V/F control is an open loop control theorem. It doesn't need the feedback of the rotating speed. Thus, to the existing motor, it can change the rotating speed just by adding the inverter without other work. The basic principle is variable frequency control. According to the rotating speed command, we adjust the

frequency of the supplying power, the same as the output frequency of the inverter, and the rotor of the electric motor changes the rotating speed. Due to the slip of the induction motor, the actual rotating speed of the motor can just close to the command of the rotating speed. If we change the output frequency without changing the supplying voltage, the torque of the motor decreases while the frequency increases. To keep the flux of the motor and the highest running efficiency, we need to adjust the output voltage; by doing so, and we can achieve the goal of keeping the flux and controlling the rotating speed.

To explain the basic theorem of V/F control, we consider the model of the single phase induction motor equivalent circuit that is shown in Fig 2.2. R_s is the stator resistance, R_r is the rotor resistance, L_m is the mutual inductance, L_{ls} is the leakage reactance of the stator, L'_{lr} is the leakage reactance of the rotor.

Considering the excitation current I_m , from the ohm's law, it can be shown as the following:

$$\frac{E}{X_m} = I_m \quad (2-2)$$

The reactance X_m can be represented as $X_m = \omega L_m$, and L_m is the mutual inductance; the above equation can be represented as follows:

$$\frac{E}{\omega L_m} = I_m \quad (2-3)$$

From the above equation, if we want to fix the ratio $E/\omega L_m$ to be constant, because of the equation $\omega = 2\pi f$, E/f must be constant, too. Generally speaking, the stator resistance is small, and it means that the voltage drop caused by the stator current can be neglected when V_s is big enough. Under the condition, the proportional constant E/f can be approximated as V_s/f .

The V/F control is an open loop control method, and it doesn't have the

accurate control performance. But it has the advantage of less cost and simple architecture.

2.2.3 Vector control theorem

Up to now, the vector control strategy and technique is used in all kinds of AC motors. The concept of the control method is equivalent to the separately excited DC motor. The separately excited DC motor, if we neglect the magnetic saturation and the armature effect, the torque is direct proportional to the armature current.

In the torque control of induction motor, the magnetic field of stator and rotor varies according to the running condition and it's not orthogonal. Through the coordinate transformation technique, we transfer the three-phase static coordinate to the two-axis (d,q) synchronous rotating coordinate system in order to make the control close to the separated DC excited motor. The current can be divided into two orthogonal, independent parts. If the flux of the rotor is fixed, the q-axis current of the stator is direct proportional to the torque.

2.2.4 The derivation of vector control

The dynamic equation of three-phase cage-type induction motor in the synchronous rotating coordinate is

$$\begin{bmatrix} R_s + L_s p & -\omega_e L_s & L_m p & -\omega_e L_m \\ \omega_e L_s & R_s + L_s p & \omega_e L_m & L_m p \\ L_m p & -(\omega_e - \omega_r) L_m & R_r + L_r p & -(\omega_e - \omega_r) L_r \\ (\omega_e - \omega_r) L_m & L_m p & (\omega_e - \omega_r) L_r & R_r + L_r p \end{bmatrix} \begin{bmatrix} i_{ds}^e \\ i_{qs}^e \\ i_{dr}^e \\ i_{qr}^e \end{bmatrix} = \begin{bmatrix} V_{ds}^e \\ V_{qs}^e \\ 0 \\ 0 \end{bmatrix} \quad (2-5)$$

p is the Laplace operator, R_s is the stator resistance, R_r is the rotor resistance, ω_e is the synchronous rotating speed, ω_r is the rotating speed of the rotor, L_s is stator inductance, L_m is the mutual inductance, the torque equation of the motor is

$$T_e = \frac{3}{2} \frac{P}{2} L_m (i_{qs}^e i_{dr}^e - i_{ds}^e i_{qr}^e) \quad (2-6)$$

The torque is produced by the vector cross product of the two axis current of the stator and rotor, it can not attain the control characteristic like the separately DC excited motor. We represent the rotor current by the rotor flux and the stator current

$$i_{dr}^e = \frac{1}{L_r} (\phi_{dr}^e - L_m i_{ds}^e) \quad (2-7)$$

$$i_{qr}^e = \frac{1}{L_r} (\phi_{qr}^e - L_m i_{qs}^e) \quad (2-8)$$

Substitute (2-7), (2-8) to (2-5), we can obtain the following matrix:

$$\begin{bmatrix} R_s + L_\sigma p & -\omega_e L_\sigma & \frac{L_m}{L_r} p & -\omega_e L_m \\ \omega_e L_s & R_s + L_s p & \omega_e L_m & L_m p \\ L_m p & -(\omega_e - \omega_r) L_m & R_r + L_r p & -(\omega_e - \omega_r) L_r \\ (\omega_e - \omega_r) L_m & L_m p & (\omega_e - \omega_r) L_r & R_r + L_r p \end{bmatrix} \begin{bmatrix} i_{ds}^e \\ i_{qs}^e \\ \phi_{dr}^e \\ \phi_{qr}^e \end{bmatrix} = \begin{bmatrix} V_{ds}^e \\ V_{qs}^e \\ 0 \\ 0 \end{bmatrix} \quad (2-9)$$

$$\text{and } L_\sigma = L_s - \frac{L_m^2}{L_r}$$

The torque equation can be rewritten:

$$T_e = \frac{3}{2} \frac{P}{2} \frac{L_m}{L_r} (i_{qs}^e \phi_{dr}^e - i_{ds}^e \phi_{qr}^e) \quad (2-10)$$

Using the state equation to represent the third and fourth row of (2-9)

$$p \begin{bmatrix} \phi_{dr}^e \\ \phi_{qr}^e \end{bmatrix} = \begin{bmatrix} -\frac{R_r}{L_r} & \omega_{sl} \\ -\omega_{sl} & -\frac{R_r}{L_r} \end{bmatrix} \begin{bmatrix} \phi_{dr}^e \\ \phi_{qr}^e \end{bmatrix} + \frac{R_r L_m}{L_r} \begin{bmatrix} i_{ds}^e \\ i_{qs}^e \end{bmatrix} \quad (2-11)$$

$$\text{and } \omega_{sl} = \omega_e - \omega_r$$

ω_{sl} is the slip frequency, to solve the steady state solution, we can get the rotor flux

$$\phi_{dr}^e = \frac{R_r L_m}{\Delta} \left(-\frac{R_r}{L_r} i_{ds}^e - \omega_{sl} i_{qs}^e \right) \quad (2-12)$$

$$\phi_{qr}^e = \frac{R_r L_m}{\Delta} \left(-\frac{R_r}{L_r} i_{qs}^e - \omega_{sl} i_{ds}^e \right) \quad (2-13)$$

$$\text{and } \Delta = \left(\frac{R_r}{L_r} \right)^2 + (\omega_{sl})^2$$

If the rotor flux falls in the direction of d axis, ϕ_{qr}^e should be zero, and the slip frequency of the motor should satisfy

$$\omega_{sl} = \frac{R_r i_{qs}^e}{L_r i_{ds}^e} \quad (2-14)$$

$$\phi_{dr}^e = L_m i_{ds}^e \quad (2-15)$$

The torque equation can be rewritten as follow

$$T_e = \frac{3}{2} \frac{P}{2} \frac{L_m^2}{L_r} i_{ds}^e i_{qs}^e \quad (2-16)$$

In the torque control of the induction motor, if the rotor flux is fixed, the torque is direct proportional to the i_{qs}^e .

The most important step of vector control is to let the direction of the rotor

flux axis the same to the direction of the synchronous d-axis.

Rotor flux rotates as the speed of electrical angular frequency ω_e . From the equation $\theta_e = \theta_r + \theta_{sl}$, $\theta_e = \int \omega_e dt = \int (\omega_r + \omega_{sl}) dt$, the synchronous rotating d-axis coincides to the direction of the rotor flux. At the time $\phi_{qr}^e = 0$, all the rotor flux falls in the direction of d-axis. When deciding the d-q axis, we can use vector control theorem through the coordinate transformation.

2.2.5 The comparison between V/f control and vector control

V/f control and vector control are the two main control theorems of the inverter nowadays. There are some disadvantages of the V/f control:

- The torque can not be adjusted in the whole speed range.
- Due to the open loop control characteristic, the dynamic response is bad.

The characteristics of the vector control are:

- Provides the constant torque in the whole speed range.
- Provides the accurate speed control owing to the closed loop control system.

2.3 The electric and mechanic system

In our treadmill system, using the inverter to control the induction motor to drive the running belt. The figure 2.3 is the mechanism of the treadmill. The induction motor drives the left side of the rotating axis; the right side of the rotating axis that has the radius R connects to the left side of the rotating axis by use of the belt. The rotating axis that has the radius r connects to the running belt.

Considers the mechanism that has the speed ratio $a=2.2/6$, f is the output frequency (Hz) of the inverter, and P is the poles of the motor, and V is the speed (m/s) of the running belt, and S is slip. Derives the following equation.

$$120 \cdot \frac{f}{P} \cdot \frac{2\pi}{60} \cdot (1-S) \cdot a \cdot R = V \cdot \frac{R}{r} / 3.6 \quad (2-17)$$

From the actual measurement, $P=4$, $R/r=3/2$, $R=0.06$ m, when the frequency of the inverter is 5.3 Hz, the linear speed of the running belt is 0.8 km/hr. substitutes the above data to the equation 2-17, the slip is 0.09. When the frequency of the inverter is 122 Hz, the linear speed of the running belt is 20 km/hr, from the equation 2-17 the slip is 0.012. From the data of the specification on the motor, the rotating speed of the motor is 1735 rpm when the frequency is 60 Hz. From the equation 2-1, the rotating speed of the motor when the frequency is 60 Hz is 1800 rpm, and we can calculate the value of slip: 0.036.

Chapter 3 Software architecture

3.1 Embedded system

Embedded system is application-oriented special computer system which is scalable on both software and hardware. It can satisfy the strict requirement of functionality, reliability, cost, volume, and power consumption of the particular application. With the rapid development of IC design and manufacture, CPUs become cheap. Lots of consumer electronics have embedded CPU and thus become embedded systems. For example, PDAs, cellphones, point-of-sale devices, VCRs, industrial robot control, or even toasters can be embedded system. There is more and more demand on the embedded system market. Some report expects that the demand on embedded CPUs is 10 times as large as general purpose PC CPUs. As applications of the embedded systems become more complex, the operating system support and development environment became crucial.

3.2 Embedded operating system

The difference in operating system between embedded system and the desktop computer is that the embedded operating system is closely relative to the application environment and needs the ability of real time performance.

The execution result of the real time operation is relative not only to the

correctness of the logic but also to the time requirement. If the real time operation can not be done before the deadline, it can not be accepted.

A real time task responses appropriately according to some events, and the event usually happens instantly. So a real time task needs to keep up with the speed of the event or it may lose the meaning of real time. A real time system is designed to complete the task within the assigning time.

The different system needs the different level of real time performance. The real time system can be divided into two kinds according to the different level of time requirement: hard real time system and soft real time system.

- Hard real time: Any task in the system must satisfy the time requirement or it may cause great damage. It's used in the condition such as weapon, aviation, and transportation, and industrial control.
- Soft real time: The system should satisfy most of the time requirement, but it can tolerate some level of latency. It's used in the condition such as mobile phone, PDA and daily entertainment.

The embedded operating system can be divided into two kinds according to the application environment: the general-purpose operating system, and the specific-purpose operating system. The typical general-purpose operating system is: WinCE.NET [8], VxWorks, Embedded Linux, and Windows XP Embedded.

And the specific-purpose operation system such as: Palm, which is used in the PDA; Symbian, which is used in the mobile phone.

When choosing the operating system, we consider the function of the treadmill and the need to the real time ability, and it is better to choose the

embedded operating system.

Considering the Microsoft and Linux embedded operating system, the advantage of the Linux embedded operating system is that the source code is open to the public. Under the open environment, many engineers can participate to test, and debug, so the stability of the operating system is good, and in the aspect of Microsoft, the early version of product inherits the drawback of the original Microsoft operating system: consumes more system resource, bad execution efficiency, and bad stability. But there is a great improvement of the new Microsoft embedded operating system. The similarity of the desktop operating system and the embedded operating system is the advantage of the Microsoft product. From the operating system to the integrate development environment, it is similar to the desktop environment, and it may take less time to familiar to the new environment. Due to the reason, we use the Microsoft embedded operating system.

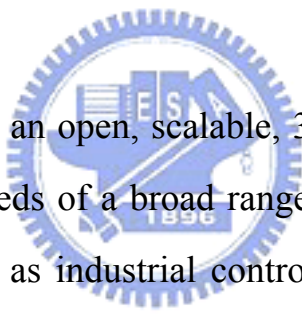
There are two Microsoft embedded operating system nowadays: WinCE.NET and WindowsXP embedded. The following is the comparison between the two embedded operating system.

- CPU architecture: WinCE.NET supports four kinds of microprocessors: ARM, MIPS, SHx, X86.
WindowsXP embedded only supports X86.
- Real time performance: WinCE.NET is the real time embedded operating system, and WindowsXP embedded needs the third party software to achieve the real time performance.

- **Memory requirement:** WinCE.NET needs at least 200 Kb and WindowsXP embedded needs at least 5 Mb. The memory requirement depends on the function of the device.

From the above discussion, WinCE.NET supports more microprocessors, and the real time performance is better, and the requirement of the memory is fewer, so we choose WinCE.NET to be the embedded operating system of our treadmill.

3.3 WinCE.NET



Windows CE .NET is an open, scalable, 32-bit operating system (OS) that is designed to meet the needs of a broad range of intelligent hardware devices, from enterprise tools such as industrial controllers, communications hubs, and point-of-sale terminals to consumer products such as cameras, Internet appliances, and interactive televisions. A typical Windows CE-based embedded platform is targeted for a specific use; often runs disconnected from other computers, and requires a small-sized OS that has a bundled, deterministic response to interrupts.

Windows CE .NET offers the application developer the ease and versatility of scripting languages, along with the versatile environment of the Microsoft Win32 application programming interface. It also offers bundled support for multimedia, Internet, LAN, and mobile communications and security services. The following are the features of Windows CE .NET.

- **Shell and User Interface**

Windows CE.NET combines the Microsoft Win32 application programming interface, user interface, and graphics device interface libraries into the Graphics, Windowing, and Events Subsystem (GWES) module. GWES is the interface between the user, the applications, and the operating system.

GWES supports all the windows, controls, and resources that make up the Windows CE UI, which enables users to control applications. GWES also includes support for user input and for GDI, which includes support for fonts, text drawing, line and shape drawing, palettes, and printing.

- **Core OS Services**

The kernel, which is represented by the Nk.exe module, is the core of the Windows CE operating system. The kernel provides the base OS functionality for any Windows CE-based device. This functionality includes process, thread, and memory management. The kernel also provides some file management functionality. The kernel services enable applications to use this core functionality. Use the kernel process and thread functions to create, terminate, and synchronize processes and threads and to schedule and suspend a thread. Processes, which represent single instances of running applications, enable users to work on more than one application at a time. Threads enable an application to perform more than one task at a time. Thread priority levels, priority inversion handling, interrupt support, and timing and scheduling are all included in the Windows CE kernel architecture.

The Windows CE kernel uses a paged virtual-memory system to manage and allocate program memory. The virtual-memory system provides contiguous blocks of memory, so that applications do not have to manage the actual memory allocation.

- **Object Store and Registry**

The object store, up to 256 MB of compressible, nonvolatile, random access memory (RAM) storage, is the default storage device on Windows CE-based platforms. The object store integrates the read-only files that are stored in read-only memory (ROM) with the read/write files of both the application and the user. In addition to the object store, Windows CE supports multiple, installable file systems, which include FAT12, FAT16, and FAT32, used by the ATA and SRAM PCCARD and Compact Flash (CF) cards, as well as proprietary systems that have their own drivers.

3.4 Development Tools

When we want to develop an application using in the embedded WinCE.NET system, the first important thing we should do is to choose one development tool. From now on, there are two main development tools for developing WinCE.NET application.

The table 3.1 summarizes two application development tools for Windows CE and the platforms and APIs they support for Microsoft Windows CE-based development.

To build Win32 applications or dynamic link libraries, we can choose Microsoft Embedded Visual C++ 4.0, for Windows CE.NET-based platforms.

To build a .NET Compact Framework application, we need Microsoft Visual Studio .NET 2003. A useful feature of this environment is that we can drag-and-drop controls from a toolbox onto a form, then click on elements in the form to add code behind the controls.

3.4.1 Win32 API

The Win32 API is the core programming interface for Windows CE. In the early 1990s, Microsoft announced that its two core strategic technologies were Win32 and the Component Object Model.

Windows CE was the first Microsoft operating system to use the Win32 API for both device drivers and applications. Sixteen-bit Windows systems use the Virtual drivers at their lowest layer, and 32-bit Windows systems have a proprietary kernel-mode API that is decidedly not Win32. Therefore, Windows CE is more deeply connected to Win32 than any other Microsoft operating system.

Win32 has been referred to as the "assembly language of Windows," because it is a very low-level API with very primitive functions. Just as multiple machine language instructions are needed to accomplish even the simplest task, multiple Win32 function calls are often needed to do real work. This is, however, the API that all other APIs and development tools ultimately rely on to get things done. In the context of the .NET Framework, Win32 code is referred to as "unmanaged code." This is because the Common Language Runtime does not manage the memory, or guarantee the security and type-safety, of Win32 code. .NET code, by contrast, is called "managed code" because all these features and more are provided by the .NET runtime.

The following are the features of the Win32 API:

- Fastest executables

Win32 provides the fastest executables. Part of the reason is that Win32 executables ship as native machine instructions. .NET Compact Framework executables, by contrast, ship as Microsoft Intermediate Language/Common

Intermediate Language (MSIL/CIL), which must be converted to native code. This conversion takes time and cannot anticipate when it occurs.

The IL-to-native conversion takes place when a page of code is decompressed from the object store and/or read-only memory (ROM) and moved to program memory. This conversion is only needed when the code is brought into program memory, so after that the code can be executed with no further conversion required, as long as it is not deleted by the system memory manager.

Another aspect of .NET Compact Framework code that can cause delays is the Garbage Collector. The Garbage Collector moves objects on the heap as part of its operation. Any managed thread that is running in a process might need to access one or more object on the heap, so to prevent this, all threads are halted. There is no question that the Garbage Collector provides a valuable service, but there is no way to schedule or control when it will run. This means that the execution of managed code might be inconsistent.

- Best real-time support

The recommendation to use Win32 for real-time support is related to its support of the fastest executables. At its core, real-time processing demands both a correct algorithm and a timely algorithm. Real-time handling is used for data collection, as well as control for devices as varied as robots in a manufacturing arena or mice and keyboards used for input.

Real-time support means more than just doing things as fast as possible. Windows CE real-time support provides a guarantee of consistency for the highest-priority thread in the system and for the highest-priority interrupt handler. Windows CE supports 256 thread priorities. It also provides the ability to manipulate the scheduling quantum of individual threads.

As described earlier, the Garbage Collector requires all managed threads to halt. But threads running in native code can continue running even when the Garbage Collector is running. If such threads return to managed code while the Garbage Collector is running, they are blocked until the Garbage Collector is finished. This means that a Win32 thread running native code can co-exist peacefully with managed code.

- Source code portability

Both Win32 and .NET Compact Framework applications provide a degree of portability: Win32 provides source-code portability, and the .NET Compact Framework provides binary portability. Win32 portability makes some source code to run on a wide range of Windows CE platforms possible, even on platforms without the .NET Compact Framework runtime. The limitations on Win32 executables is that they rely on having the necessary Win32 functions and are CPU-dependent.

Windows CE is a highly configurable operating system. This means that the set of Win32 functions supported on one platform might not match the set of Win32 functions on a second platform. Even when using the Win32 API, getting the portability to a broad range of platforms requires some diligence.

- Ability to create device drivers

All device drivers should be written using Win32. Some of the reasons have already been mentioned: size, speed, real-time support, and portability to the broadest range of platforms.

3.4.2 .NET Compact Framework

Whereas Win32 is good at creating low-level code for device and operating system support, the .NET Compact Framework is good for interacting at a high level. It is best suited for building an application with a user interface that collects data, stores it in a local database, and, from time to time, forwards that data to a server-based database.

As mentioned earlier, Win32 code is referred to as unmanaged code and .NET code is called managed code. The term "managed code" refers to the fact that the Common Language Runtime (CLR), which might also be called the "code manager," provides several assurances for such code:

- Managed code cannot have bad pointers.
- Managed code cannot create memory leaks.
- Managed code supports strong type-safety.

The following are some conditions to use .NET Compact Framework:

- Using platforms that have the .NET Compact Framework

To be able to run .NET Compact Framework applications, platforms must support the .NET Compact Framework runtime. A second requirement for running .NET Compact Framework applications is that platforms must support a sufficient subset of the Win32 API, the foundation on which the .NET Compact Framework was built. The set of Win32 APIs that are required to support .NET Compact Framework must be present. .NET Compact Framework cannot run on headless Windows CE.NET-based platforms, which include Media Appliance, and Residential Gateway. To run the .NET Compact Framework, a Windows CE .NET platform must have a display screen.

- Building user interfaces

The .NET Compact Framework does a very good job of building a user

interface. The Visual Studio .NET Forms Designer is allowed to drag and drop controls from the toolbox onto a form. The Properties window helps identify supported events and properties for the various controls, all from a straightforward user interface.

But there are differences. Although 28 of the 35 desktop controls are supported in the .NET Compact Framework, each control has been refined to meet the size and performance requirements of Windows CE. For this reason, a subset of the desktop Properties, Methods, and Events are supported in the .NET Compact Framework controls. The biggest impact will be on code that writes on the desktop and ports to the .NET Compact Framework.

- Achieving binary portability to multiple CPUs

On a platform with .NET Compact Framework, a single .NET Compact Framework executable file will run on multiple CPUs. The key benefit here is that it simplifies the setup process when targeting platforms that support multiple CPUs. This will be particularly helpful for .NET Compact Framework libraries with custom controls and other useful, multi-platform widgets.

- Using existing .NET Framework code

Another reason to adopt the .NET Compact Framework is having desktop .NET Framework code. The .NET Compact Framework is a rich subset of the desktop .NET Framework, and the key elements are the same, including namespaces, classes, method names, property names, and data types.

However, the .NET Compact Framework is a much smaller library than the desktop .NET Framework. Comparing just the binary files, the

desktop .NET Framework is approximately 30 megabytes, while the .NET Compact Framework is 1.5 megabytes. It represents a subset of the desktop .NET Framework that is finely tuned for both size and performance.

3.5 Platform Invoke (P/Invoke)

.NET Compact Framework supports the P/Invoke service. This service allows managed code to invoke unmanaged functions residing in DLLs. Although the .NET Compact Framework supports P/Invoke, it does so a little differently than the full .NET Framework.

Just as in the .NET Framework, using the P/Invoke service in the .NET Compact Framework includes three primary steps: declaration, invocation, and error handling.

- Declaration

To make the .NET Compact Framework know which unmanaged function that intends to call. We need to include the DLL name, the function name and the calling convention to use.

- Invocation

When correctly declared the function to call, and wraps the function call in a method of that class. The normal technique used is to declare the wrapper function as a Public static method of the class. When this code is just-in-time (JIT) compiled at runtime, the P/Invoke service of the common language runtime will extract the `Declare` or `DllImportAttribute` definition from the metadata in the assembly, locate and load the DLL containing the function into memory, and then retrieve the address of the function using

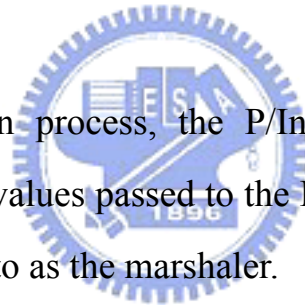
the entry point information. If all goes well, the function will then be invoked, its arguments marshaled, and any return values passed back to the caller.

- **Handling Errors**

Using P/Invoke can generate errors such as exception generated by the P/Invoke service itself. This occurs if the arguments passed to the method contain invalid data, or if the function itself is declared with improper arguments.

3.5.1 Marshaling Data

During the invocation process, the P/Invoke service is responsible for marshaling the parameter values passed to the DLL function. This component of P/Invoke is often referred to as the marshaler.



3.6 ActiveSync

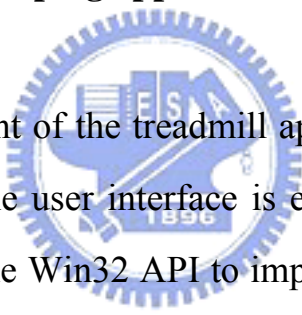
Microsoft ActiveSync provides support for synchronizing data between a Windows-based desktop computer and Microsoft Windows CE-based devices.

There are many choices of the connection between the desktop computer and the mobile device such as: RS-232, Ethernet, and USB.

3.7 Platform Builder

Platform Builder is an integrated development environment (IDE) for building customized embedded platforms based on the Windows CE.NET operating system. Platform Builder comes with all the development tools necessary to design, create, build, test, and debug a Windows CE-based platform. The IDE provides a single integrated workspace in which can work on both platforms and projects.

3.8 The procedure of developing application



Due to the development of the treadmill application, we use C# [10][11] to develop the application, the user interface is easier to implement. And we can use P/Invoke [13] to use the Win32 API to implement the rs232 communication and the play of audio data. When we complete the application, we need to run the application to debug; there are two ways to debug: uses the emulator and downloads the application to test. Using emulator is convenient when there is no development board, but there may be some different to the real situation. Downloading the application to the development board has the following procedures:

- (1) Establishes the new connection in WinCE.NET.” My Computer→Control Panel→Network and Dial-up Connections→Make New Connection”.
- (2) Chooses “Direct Connect” in the “Make New Connection”, as shown in figure 3.1 “Choose Direct Connection→Connection on COM1 (name)”.
- (3) Choose serial communication connection. “Select a device→COM1→

Finish”. It is shown in fig 3.2.

- (4) Installs ActiveSync in the development computer.
- (5) In the development computer, opens the ActiveSync, “File→Get Connected→Next”. The picture of ActiveSync is shown in fig 3.3.
- (6) In the target, types the “repllog” in “Run” can make the connection valid.



Chapter 4 Implementation

4.1 Architecture

The whole architecture of the treadmill can be shown in Fig 4.1. The whole architecture can be briefly described as follows. In the treadmill system, the user can use the treadmill through the 15 inch touch screen. The controller of the touch screen to communicate with the target board PCM-9575 is through USB interface. The connection between the desktop pc using to develop the application and the target embedded development board PCM-9575 is through the RS-232 interface, and we need the software ActiveSync to make the connection valid.

The development board PCM-9575 communicates with the Inverter EI-8001 through the RS-232 interface to control the AC induction motor. We can play the music or sound with the help of stereo speaker, and download music file through the Ethernet.

4.2 Communication between PCM-9575 and Inverter EI 8001

There are two choices to control the Inverter EI-8001: terminal control, and serial communication control.

4.2.1 Terminal control

Most treadmills use the terminal control to control the speed of the motor, figure is the graph of the terminal, and table is the function of each pin. The pin AI1 controls the voltage to control the speed. The pin LI1, LI2, LI3, LI4 can be programmed to the required function such as free run.

4.2.2 Serial communication control

EI-8001 adopts RS-232, half-duplex, 9600 baud rate, no parity, and 8 data bits. The communication between the development board PCM-9575 and the inverter is:



PCM-9575->Inverter (command)

2Fh	Drv.-No.	Cmd.	Log-Adr.	Data	Sum
-----	----------	------	----------	------	-----

Inverter->PCM-9575 (answer)

2Fh	Drv.-No.	Cmd.	Log-Adr.	Data	Sum
-----	----------	------	----------	------	-----

Inverter->PCM-9575 (wrong message)

2Fh	Drv.-No.	4Eh,6Eh	Data	Sum
-----	----------	---------	------	-----

The following are the meanings of each block:

2Fh ('/') [1 byte]: Heading byte

Drv.-No. [1byte]: When the value of the Drv.-No. matches the internal drive number, the command will execute, the default value is 0

CMD [1 byte]: 52h ('R'): RAM data read. Reads the important parameters from the inverter.

 57h ('W'): EEPROM/RAM data write. Writes data to both EEPROM and RAM.

 50h ('P'): RAM data write. Writes data to RAM only.

Log.-Adr. [2 bytes]: Logical address

DATA [2 bytes]: Data to write

SUM [1 byte]: Check sum

The PCM-9575 sends command to the inverter, if the inverter receives the correct command, it returns the answer else it returns wrong message, the data in the wrong message stream is the error code, and the following is the meaning of each error code.

Error code

0000h: can't execute the command.

0001h: error data (out of range)

0002h: error register address

0004h: Checksum Error

No answer: error format



When using the Inverter EI-8001 to drive the motor, there are several important register addresses that provide the useful function. The important register address is shown in table 4.1

The logical address of FA00 is the line command register; the table 4.2 is the communication command word bit structure.

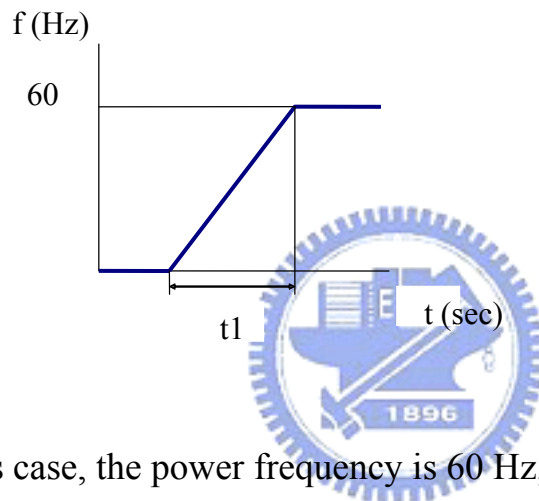
4.2.3 Acceleration time, deceleration time

The range of the acceleration time and the deceleration time is 0.1~3600 sec, its scale factor is 0.1, and the definition is based on the supplying power frequency.

For example:

Power freq. =60 Hz

Acceleration time (t_1) =10 s



In this case, the power frequency is 60 Hz, and the setting of the acceleration time is 10 sec, so increasing the frequency from 0 Hz to 30 Hz will spend 5 sec.

The acceleration time and deceleration time we use in the treadmill is 15.6 second; it can stop the machine from 3.2 km/hr to 0 km/hr in 5 second.

4.2.4 Terminal control v.s serial communication control

Most of treadmills use terminal control, because the program that previously controls the DC motor can be used directly to control the inverter, but the terminal control provides less function than the serial communication control,

and it may restrict the function when programming. So we adopt serial communication control to control the inverter.

4.2.5 The process of start and stop

The process of start and stop is important to the treadmill; the appropriate design will make the user exercise in the safety condition and prevent the occurrence of injury.

Start:

The process of start is shown as follows:

- (1) Lower the incline to 0%
- (2) Count down 3 second with the warning sound.

Stop:

There are four kinds of stop type, all of which can be described as follows:

- Cool down:

This type is used when the user presses the stop button, the standard process is: (1) Lower the incline to 0%. (2) According to the recent speed, if the speed is slower than 3.2 km/hr, then stops the treadmill in 5 sec, else if the speed is faster than 3.2 km/hr, decreasing the speed at the rate of 0.2 km/hr per second until reaching the speed of 3.2 km/hr, in this period, if the user presses the fast button, the treadmill will hold the present speed. Else if the speed is smaller than 3.2 km/hr, the treadmill will stop in 5 second.

- Pause:

This type is used when the user presses the pause button, and the

treadmill will stop according to the deceleration time.

- **Emergency Stop:**

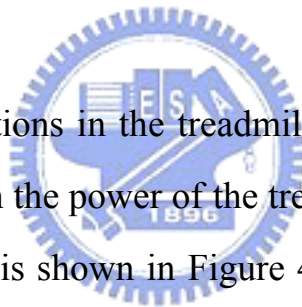
This type is used when the emergency condition occurs, and the treadmill will stop depends on the inertia and the braking ability of the drive.

- **End of course:**

This type occurs when the course or program ends, and the treadmill will stop the same as the Cool Down type.

4.3 The Function of the treadmill

There are many functions in the treadmill, and the rough program flow is shown in Figure 4.2. When the power of the treadmill is on, it gets into the Main layer; the picture of Main is shown in Figure 4.3. We can set the parameters in Data Setting; the picture is shown in Figure 4.4; we can play music in the Music Mode, the picture is shown in Figure 4.5.



4.3.1 Mode of the treadmill

- **Basic Mode:**

This mode is designed for the user who wants to use the basic function of the treadmill, and the user can run or walk at the speed between 0.5 km/hr to 20 km/hr or adjust the incline to climb at the range of 0 % to 15%. At this mode, we can adjust the speed by use of “Fast” and “Slow” button, which can adjust 0.1

km/hr when pressing the button each time. Using “hot key” is another way to adjust the speed quickly, hot key is the button used to adjust the speed quickly, and there are many “hot key” buttons: 2, 4, 6, 8, 10, 12, 14, 16, 18 km/hr. And the “Up”, “Down” button can adjust the incline 1%/press. Figure 4.6 is the picture of Basic Mode.

- SET

This mode can be divided into three parts:

(1) Set Time

This mode provides the user to set the desired exercising time; the range of the time setting is between 1 minute to 98 minute, and the adjustment unit is minute. At the end of the count down, the screen will display “End”, and the treadmill will cool down. Figure 4.7 is the picture of Set Time.



(2) Set Distance

This mode provides the user to set the desired exercising distances; the range of distance setting is between 1 km to 50 km, and the adjustment unit is km. At the end of the count down, the screen will display “End”, and the treadmill will cool down.

(3) Set Calories

This mode provides the users to set the desired consuming calories; the range of setting is between 40 cal to 990 cal, and the adjustment unit is 10 cal. At the end of the count down, the screen will display “End”, and the treadmill will cool down.

- Program

There are three different kinds of programs the users can choose to exercise: Walking, Running, Interval, each course has ten sections, and each

section takes one minute. When the tenth section is through, it returns to the first section and continue the course. The incline is fixed, and the speed can be adjusted by the user. The range of the speed is between 0.5~20 km/hr. The incline of each course and level can be shown in Table 4.3~Table 4.5. Figure 4.8 is the figure of Walking Course.

- Course

There are four different kinds of courses the user can choose to exercise:

(1) Cardio 20 Min:

There are seven sections in the course, and the first section is the warm up section, and the last section is the cool down section; the second to sixth section is the cardio section. The speed and incline of the first and last section are not open to the user; it will run by the default value. At the cardio section, there also have default values of incline and speed, but the value can be changed by the user. At the end of one cardio section, the value of speed and incline will return to the next section's default value.

There are three minutes in the warm up section, the speed of the first minute is 3.2 km/hr, and the second minute is 4.0 km/hr, and the third minute is 4.8 km/hr. There are two minutes at the cool down section; the speed slows at the rate of 0.2 kph/ 15 sec until 3.2 km/hr and then stops the treadmill in five second. The Table of cardio course is shown in Table 4.6. The figure 4.9 is the picture of CARDIO 20 Min.

(2) Interval 25 Min:

There are eleven sections in the course, and the first section is the warm up section, and the last section is the cool down section; the second to tenth section is the interval section. The speed and incline of the first

and last section are not open to the user; it will run by the default value. At the Interval section, there also have default values of incline and speed, but the value can be changed by the user. At the end of one interval section, the value of speed and incline will return to the next section's default value.

There are three minutes in the warm up section, the speed of the first minute is 3.2 km/hr, and the second minute is 4.0 km/hr, and the third minute is 4.8 km/hr. There are two minutes at the cool down section; the speed slows at the rate of 0.2 kph/ 15 second until 3.2 km/hr and then stops the treadmill in five second. The data of interval course is shown in Table 4.7.

(3) Fat Burn 30 Min:

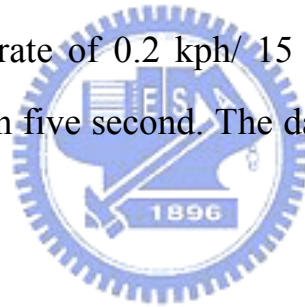
There are thirteen sections in the course, and the first section is the warm up section, and the last section is the cool down section; the second to twelve section is the Fat Burn section. The speed and incline of the first and last section are not open to the user; it will run by the default value. At the Fat Burn section, there also have default values of incline and speed, but the value can be changed by the user. At the end of one Fat Burn section, the values of speed and incline will return to the next section's default value.

There are three minutes in the warm up section, the speed of the first minute is 3.2 km/hr, and the second minute is 4.0 km/hr, and the third minute is 4.8 km/hr. There are two minutes at the cool down section; the speed slows at the rate of 0.2 kph/ 15 second until 3.2 km/hr and then stops the treadmill in five second. The data of Fat Burn course is shown in Table 4.8.

(4) Endurance 35 Min:

There are fifteenth sections in the course, and the first section is the warm up section, and the last section is the cool down section; the second to fourteenth section is the Endurance section. The speed and incline of the first and last section are not open to the user; it will run by the default value. At the Endurance section, there also have default values of incline and speed, but the value can be changed by the user. At the end of one section, the value of speed and incline will return to its default value.

There are three minutes in the warm up section, the speed of the first minute is 3.2 km/hr, and the second minute is 4.0 km/hr, and the third minute is 4.8 km/hr. There are two minutes at the cool down section; the speed slows at the rate of 0.2 kph/ 15 second until 3.2 km/hr and then stops the treadmill in five second. The data of Endurance course is shown in Table 4.9.



● Race Mode

In this mode, the user has a contest with an opponent. And there are many racing distance the user can choose: 400m, 800m, 1600m, 3000m, and 5000m. There are three levels the user can choose: Easy, Normal, and Professional. In the different level, the opponent will have different speed. In the level “Easy”, the opponent has the speed at the range between 3 km/hr~9 km/hr. In the level “Normal”, the opponent has the speed at the range between 6 km/hr~12 km/hr. In the level “Pro.”, the opponent has the speed at the range between 9 km/hr~15 km/hr. In each level, the opponent changes their speed every five second. At the end of the race, the panel will show the racing result. The figure 4.10 is the picture of RACE mode.

4.3.2 The importance of the Warm Up and Cool Down

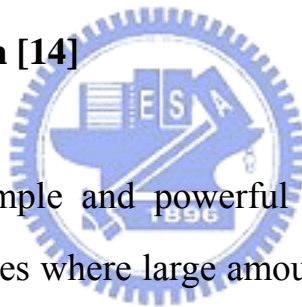
- Warm Up

First several minutes of a workout should be devoted to warm up. The warm up will limber your muscles and prepare them for more strenuous exercise. Warm up on the treadmill by walking at slow speed.

- Cool Down

A cool down period allows the heart to readjust to the decreased demand. Use a slow speed setting during the cool down to gradually the heart rate.

4.4 HTTP communication [14]



HTTP provides a simple and powerful communications mechanism for mobile applications. In cases where large amounts of data need to be transferred or data is more efficiently transferred in binary form (such as MP3s or bitmaps), it may be more efficient to take direct control of the HTTP communications process. This is especially true for mobile applications that are often faced with low bandwidth connectivity or communication billing rates based on the number of megabytes transferred. In these situations, operational performance and costs can be improved by transferring only the core application data.

The .NET Compact Framework provides two HTTP communications classes: `HttpWebRequest` and `HttpWebResponse`. Together, these classes provide the necessary functionality to both send an HTTP request (`HttpWebRequest`) to a web server and receive a response (`HttpWebResponse`) back from that server.

In the other aspect, if we want to use the download program; we need a

virtual directory on a server running IIS. And from the HTTP communication, we can download the file from the IIS server.

4.5 Play the wave file format [15]

In the treadmill system, we want to play music or sound on the treadmill. The audio data we want to play is the wave file format, using the P/Invoke to implement the mechanism. The following functions and structure are used to play waveform audio:

`waveOutGetNumDevs ()`: Determines the number of audio drivers available for output.

`waveOutOpen ()`: Creates an instance of the specified audio device for output.

`waveOutGetVolume ()`: Returns the volume of the specified output device.

`waveOutSetVolume ()`: Sets the volume of the specified output device.

`waveOutPrepareHeader ()`: Prepares a WAVEHDR and data block for output.

`waveOutUnprepareHeader ()`: Releases a previously prepared WAVEHDR and data block.

`waveOutWrite ()`: Starts playing the queued buffer.

`waveOutClose ()`: Closes the specified instance of the audio device.

`waveOutReset ()`: Stops playing and empties the queue.

`waveOutPause ()`: Pauses playback.

`waveOutRestart ()`: Resumes paused playback.

4.5.1 The Wave File Format

The Waveform Audio Interface provides a mechanism for playing and recording raw uncompressed pulse code modulated (PCM) data. The .wav file format consists of some header information followed by the raw data. This format is shown in Figure 4.11.

To reduce memory use, we can first specify a buffer size. And the audio file will be streamed using two audio buffers, each being half of the specified total size. The streaming mechanism is maintained by loading one buffer while the other plays. Therefore it is important to always have a second buffer queued before the first ends. The Figure 4.12 briefly describes the method.

The application must then wait for a message from the audio system indicating that an audio block has finished, at which point it will determine what to do next. If the audio block is the last one then playback is stopped and resources are cleaned up, otherwise, the buffer is reused to load the next block while the other buffer is playing.

The general process for creating an instance of WaveOut and playing the .wav file is as follows:

- (1) Get the number of output audio devices
- (2) Enumerate the devices and select the appropriate one
- (3) Open the audio device
- (4) Load audio data into WAVEHDR instances.
- (5) Prepare the headers
- (6) Write the headers to the audio device (put them in the queue for playback)

4.6 GDI

Due to the resource constraints, there is no full GDI support in the .NET Compact Framework. The services of GDI fall into the following three broad categories:

- (1) 2-D vector graphics
- (2) Imaging
- (3) Typography

Of the 2-D Vector Graphics in the .NET Compact Framework, only the core drawing primitives like Ellipse, Line, Image, Polygon, Rectangle, String, Fill Ellipse, Fill Polygon, Fill Rectangle, and Fill Region are supported by the Graphics object.

In the user interface of the treadmill, we use the technique to draw the playground and the incline graph and make the image buttons [16] to implement the graphic.

Chapter 5 Conclusion

From the implementation of the treadmill, we may do some improvement compared to the traditional treadmill such as user interface and the play of the audio sound and the connection to the internet. It's the advantage of using embedded system as the electric control system, but there are still some functions that haven't implement yet. Such as saving the files of some parameters in the memory, and the interface of Emergency Stop; in the traditional treadmill, the interface of Emergency Stop is done by hardware. But in our treadmill, we implement it by the software emulation. It's not a safety way by doing so. And the measurement of the Heart Rate is another function that we should implement.

In the other aspect, the use of the embedded OS, WinCE.NET has some topics for discussion: The embedded system is used for the specified purpose, and the hardware is differ from each other according to the manufacturer, so the driver for the hardware may not open to the public, and it may increases the cost. Building the WinCE.NET OS needs the driver. We should take this into consideration.

In the aspect of playing music, the format we use now is the PCM data, and it is uncompressed format. It takes much memory space, and it is improper to the embedded system; it can be improved by using the compressed format. In the aspect of drawing the playground and incline graphic, we use the basic geometric graph to form the drawing; it's better than the traditional LED display but still can be improving.

Reference

- [1] ANTHONY J. PANSINI, "Basics of Electric Motors", PRENTICE HALL, 1989.
- [2] Denis O'Kelly, "Performance and control of Electrical Machines", McGraw-Hill, 1991.
- [3] P.C. Sen, "Principles of Electric Machines and Power Electronics", John Wiley & Sons, 1997.
- [4] 劉昌煥, "電機機械", 東華, 民國 91 年.
- [5] 劉昌煥, "交流電機控制-向量控制與直接轉矩控制原理", 東華, 民國 92 年
- [6] 探矽工作室, "2002 嵌入式系統開發聖經", 學貫, 民國 91 年.
- [7] M. Morris, "Computer System Architecture", Prentice-Hall, 1994.
- [8] 黃泰一, "Windows CE 嵌入式系統理論與實務", 文魁, 民國 93 年.
- [9] Embedded System Design, "Frank Vahid", John Wiley & Sons, 2002.
- [10] 呂文達, "C#範例精要解析", 文魁, 民國 92 年.
- [11] 孫三才, "C#與.net Framework 實戰演練" 學貫, 民國 93 年
- [12] Douglas Boling, "Programming Microsoft Windows CE.NET", 3rd Edition, Microsoft Press, 2003.
- [13] Chris Tacke, "P/Invoking Serial APIs in the Compact Framework", MSDN library, 2003
- [14] Jim Wilson, "Improving .NET Compact Framework HTTP

Communications using HttpWebRequest and Custom ASP.NET Providers”,
MSDN library, 2003.

[15] Seth Demsey, ” Recording and Playing Sound with the Waveform Audio
Interface”, MSDN library, 2004.

[16] Alex Yakhnin, ” How to Create a Microsoft .NET Compact
Framework-based Image Button”, MSDN library, 2003.

[17] 江高舉, ”PhotoImpact 10 私房書”, 志凌, 民國 94 年.



List of Figures

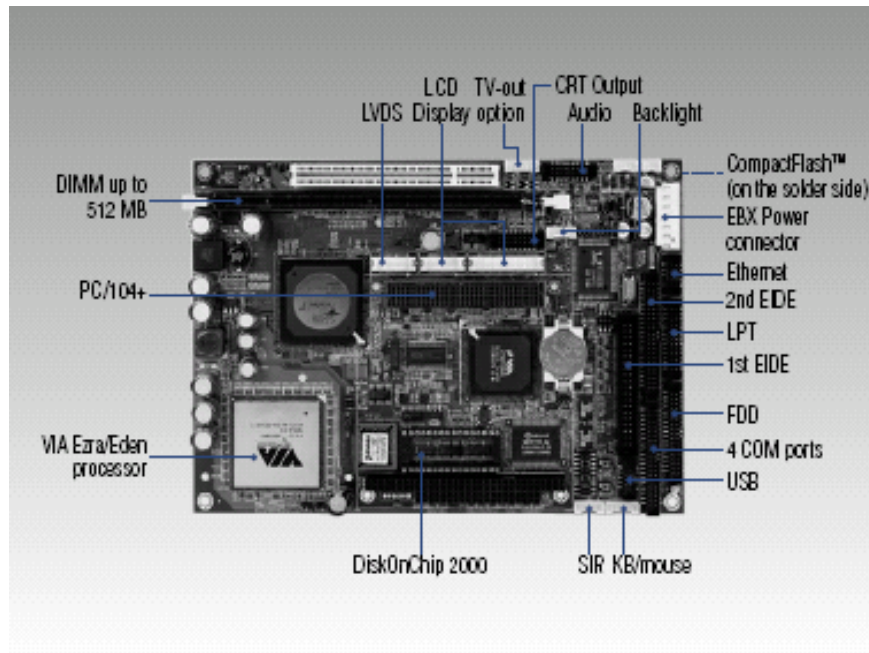


Figure 2.1 PCM-9575

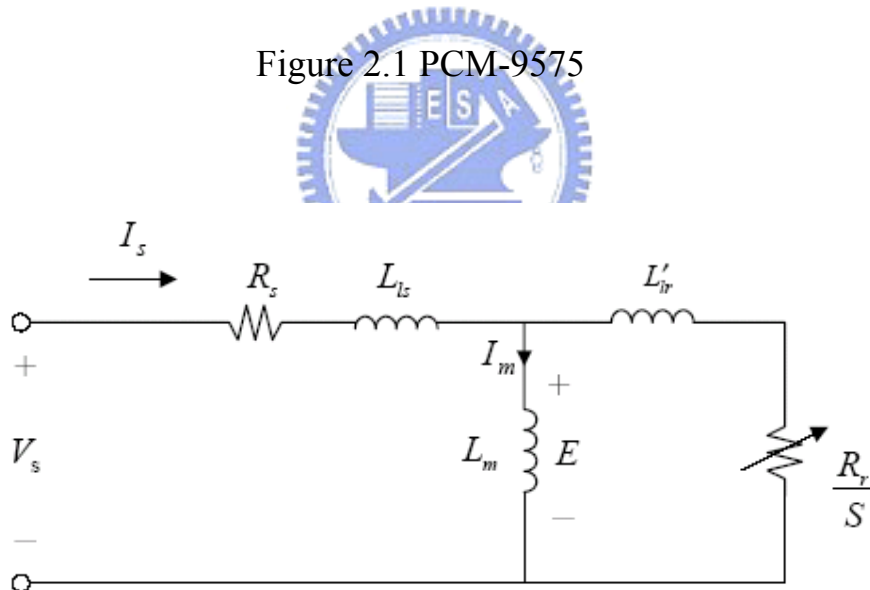


Figure 2.2 The equivalent circuit of single phase induction motor

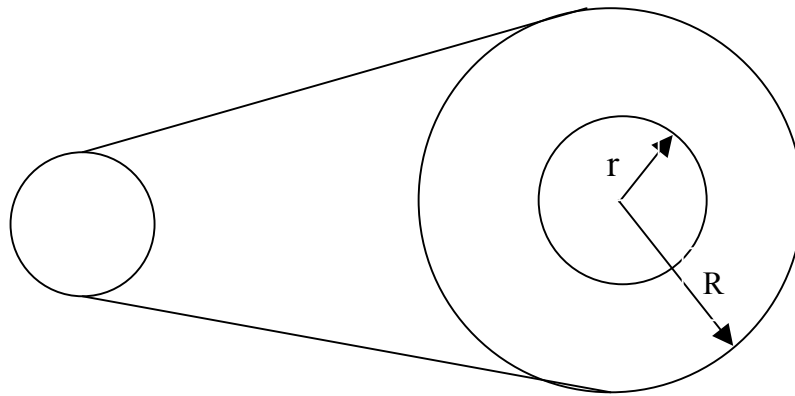


Figure 2.3 Mechanism of the treadmill

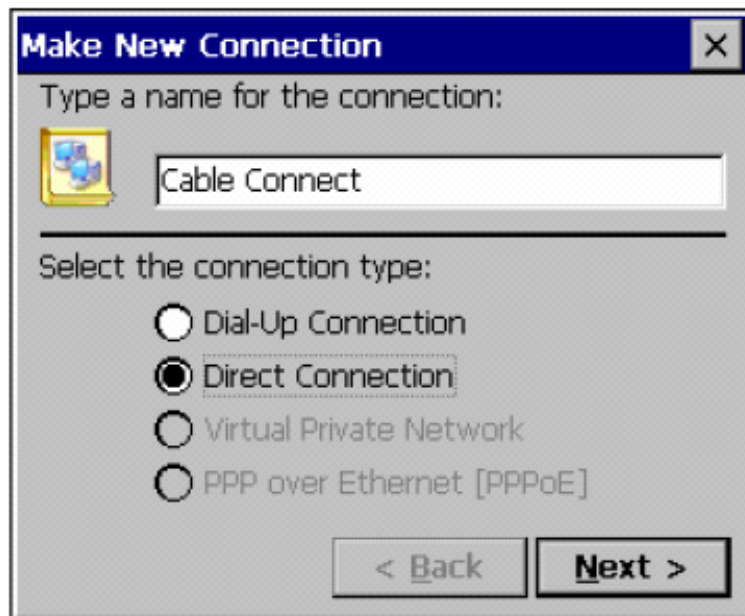


Figure 3.1 Direct Connection



Figure 3.2 Connect type



Figure 3.3 Picture of ActiveSync

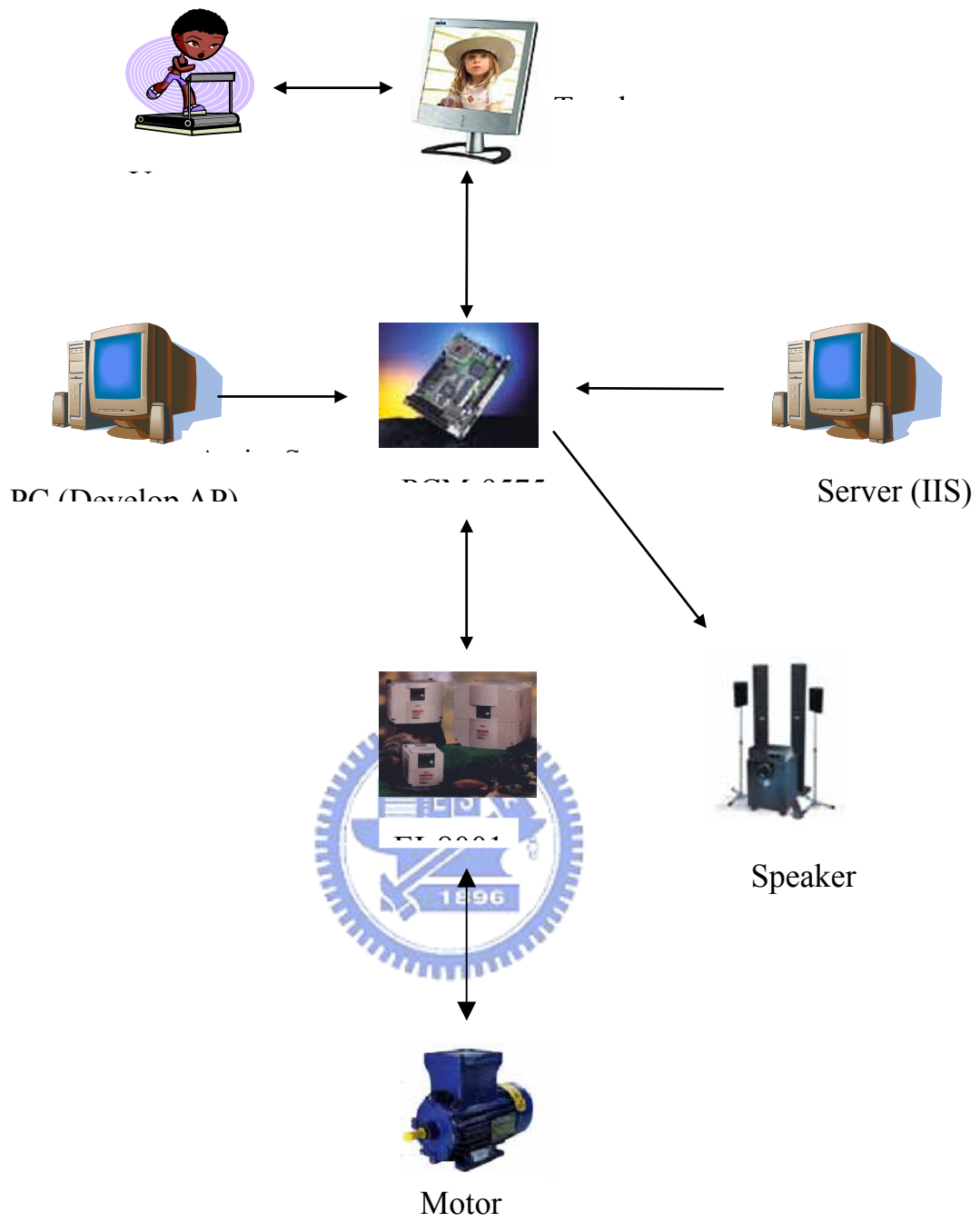


Figure 4.1 The whole architecture of the treadmill

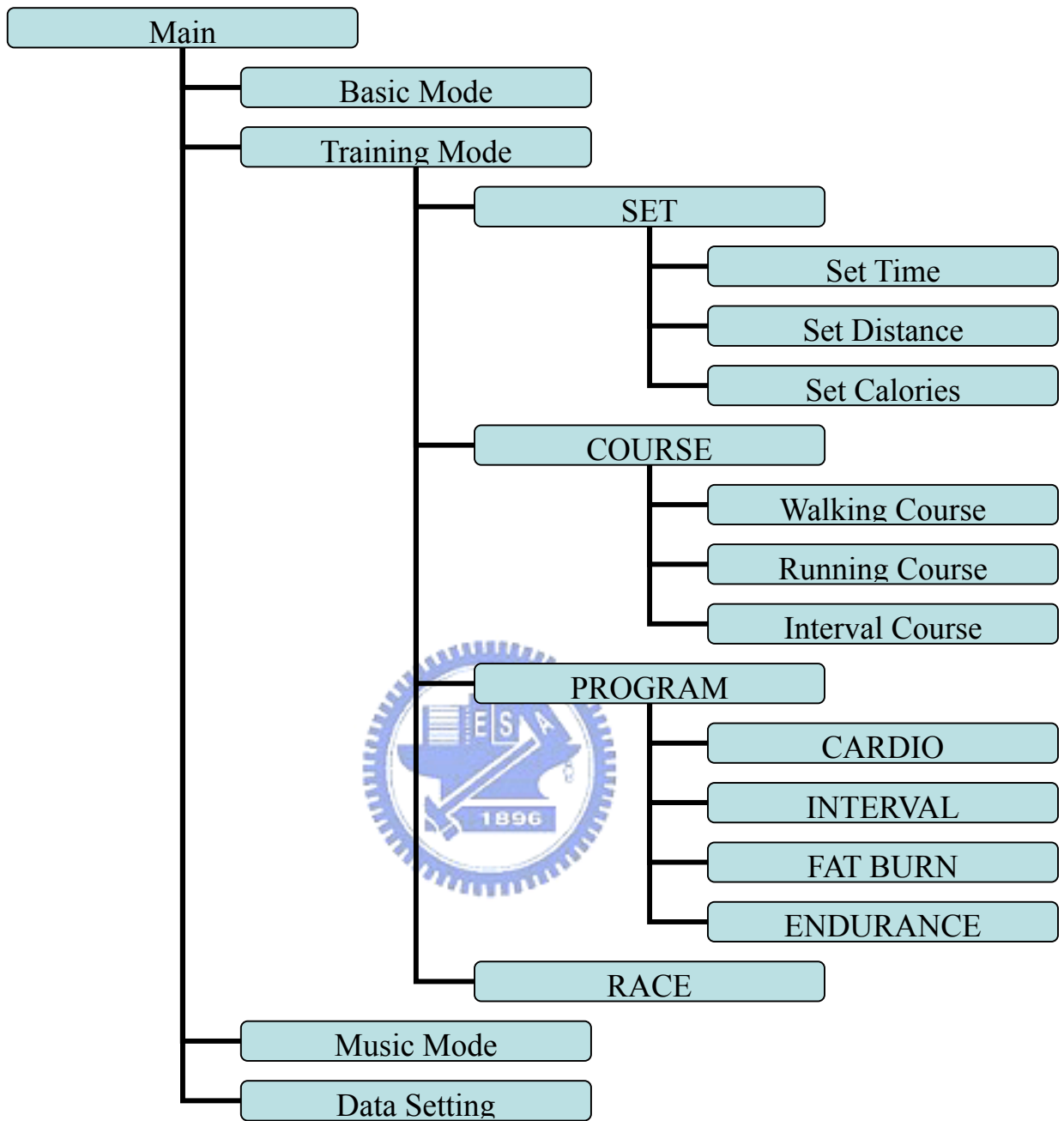


Figure 4.2 The program flow of the treadmill



Figure 4.3 Main (function)



Figure 4.4 Data Setting (function)

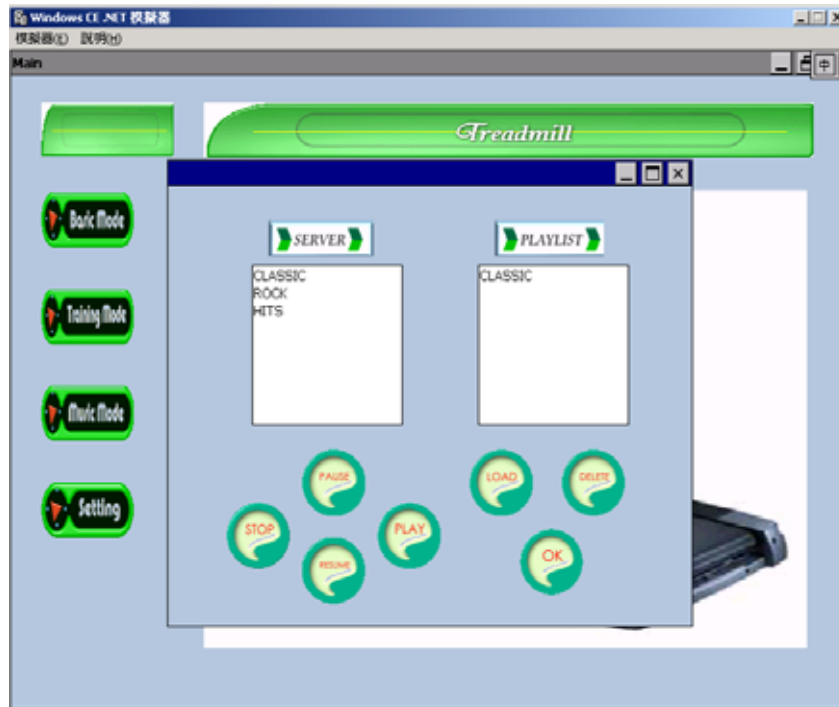


Figure 4.5 Music Mode (function)

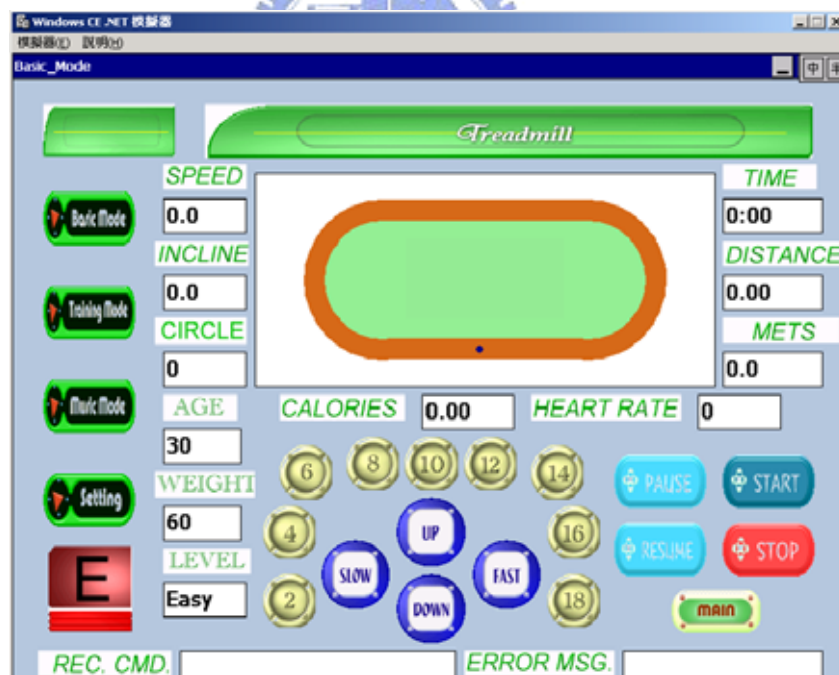


Figure 4.6 Basic Mode (function)

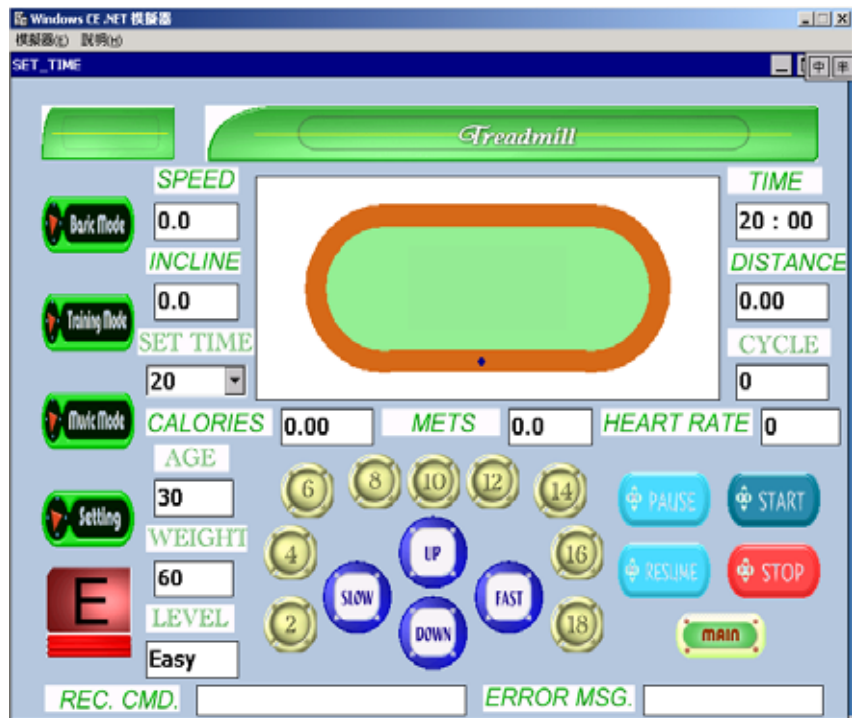


Figure 4.7 Set Time (function)



Figure 4.8 Walking Course (function)

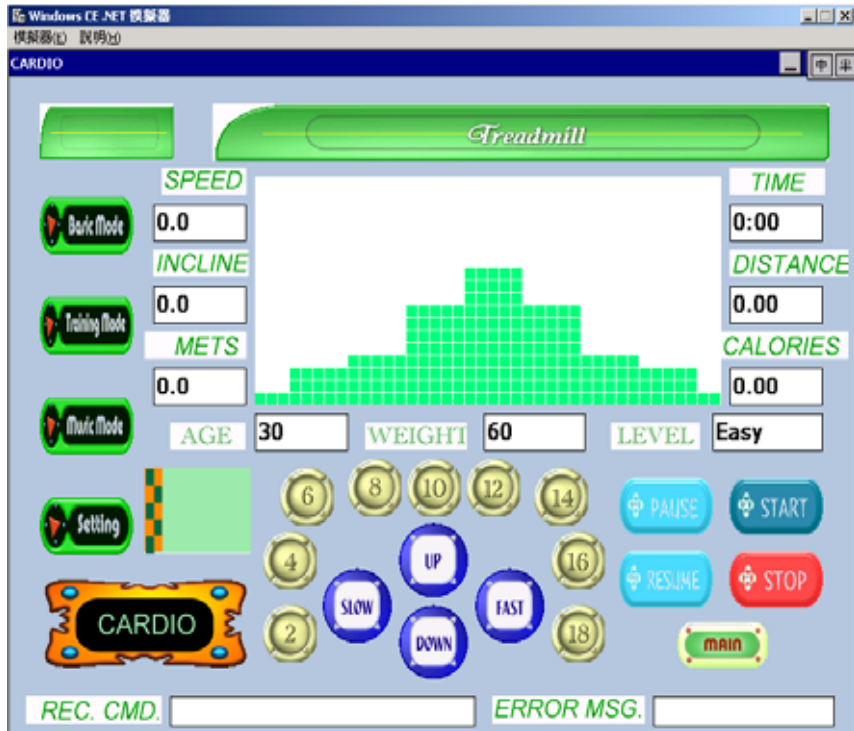


Figure 4.9 Cardio (function)



Figure 4.10 Race (function)

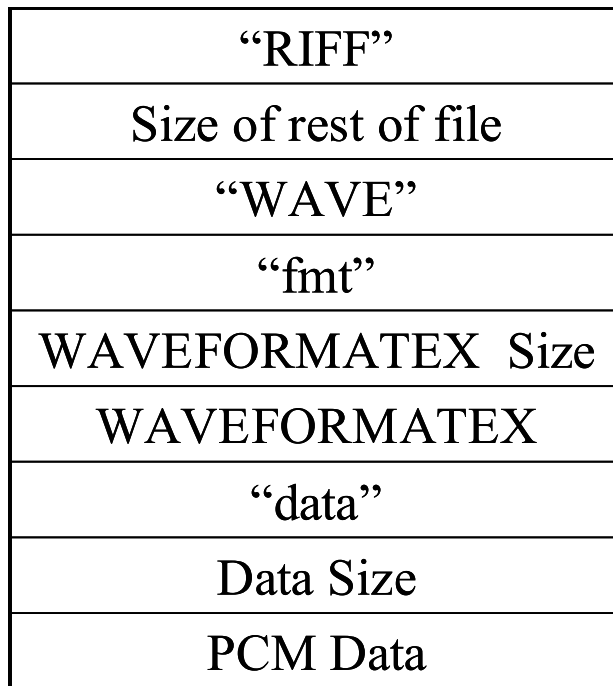


Fig 4.11 The wave file format

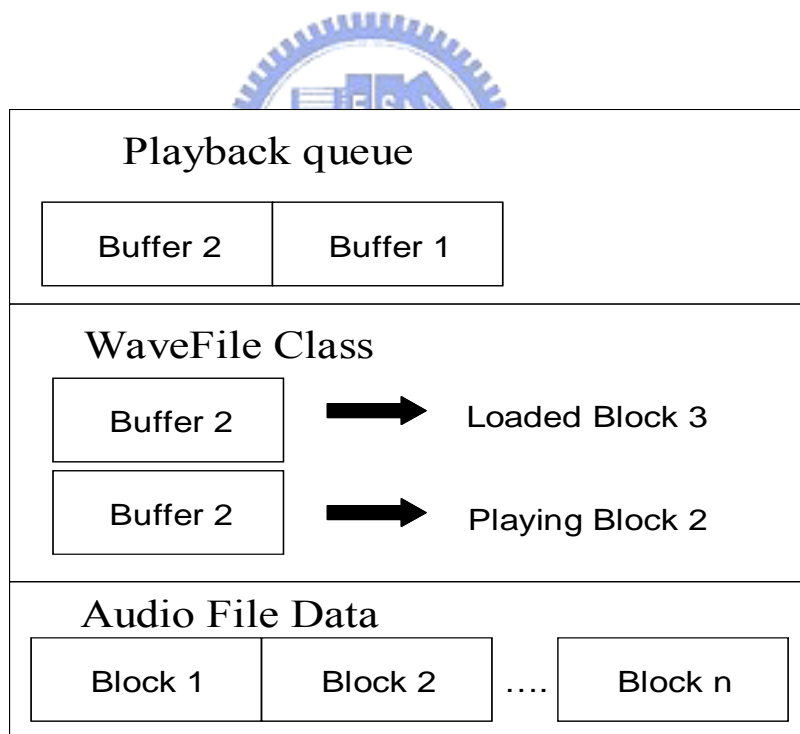


Fig 4.12 The mechanism of playing .wav file

Lit of Tables

Model NO.	CPU	Cache	Display Chipset
PCM-9575	Eden-667 MHz (○)	L1 128KB L2 64KB (○)	VIA Twister chip with integrateds3 savage4 2D/3D/Video Accelerator (○)
PCM-9572	P3-500MHz ()	L2 256KB ()	2X AGP Lynx 721 chip, 3D Engine (○)
PCM-9550	P-266MHz (X)	L2 512KB SRAM(○)	Asiliant 69000 ()
PCM-9372	Eden-400 MHz ()	L1 128 KB L2 64 KB ()	VT8606 Twister chip with integrated S3 savage4 2D/3D/Video Accelerator ()
PCM-9371	Celeron-400MHz ()	L2 256KB ()	VT8606 Twister chip with integrated S3 savage4 2D/3D/Video Accelerator ()
PCM-9340	Elite-133 MHz (X)	N/A (X)	LynxEM+712 chip (X)

Table 2.1 The comparison of development board

CPU	Embedded VIA low power Eden processor, On-die 128 KB L1 cache memory, Eden-667
2 nd Cache Memory	64 KB on the Eden processor
System Chipset	VT8606, VT82C686B, 133 MHz FSB
BIOS	AWARD 256 kbit Flash BIOS
System Memory	SDRAM 168-pin DIMM x 1, Max: 512 MB PC133 DIMM for all models,
SSD	CompactFlash card Type I/II and DiskOnChip
Expansion Interface	PC/104-Plus
Chipset	VIA Twister chip with Integrated S3 Savage4 2D/3D/Video Accelerator
Memory Size	8/16/32 MB frame buffer using system memory
Resolution	1024 x 768
LCD Interface	4x AGP VGA/LCD interface

Table 2.2 The specification of PCM-9575

Development Tool	Platform	API
Embedded Visual C++ 4.0	Windows CE .NET-based platforms	Win32 MFC ATL
Visual Studio .NET 2003	Pocket PC Pocket PC 2002 Windows CE .NET-based platforms	.NET Compact Framework


Table 3.1 Comparison of the development tool

logical address	Function
FA00h	Communication cmd. valid
FA01h	Frequency cmd.
FE01h	Drive status
0002h	Output current
0003h	Rotation frequency
0006h	Acceleration time
0007h	Deceleration time
0008h	Lower limit frequency
0009h	Upper limit frequency
0025h	Control mode
0028h	Base frequency

Table 4.1 The important register addresses of Inverter EI-8001

Bit	Function	0	1
15	FA00 valid	invalid	valid
14	FA01 valid	invalid	valid
13	Reset trip	OFF	reset
12	Emergency stop	OFF	Fast stop
11	Free run command	OFF	Free run
10	Run/Stop	Stop	Run
9	Forward/Reverse	Forward	Reverse
8	Jog operation	Off	Jog
7	DC braking	Off	DC braking

Table 4.2 The line command register of FA00



Level		1	2	3	4	5	6	7	8	9	10
Time (min)		1	1	1	1	1	1	1	1	1	1
Incline(%)	Easy	0	2	6	1	4	8	3	0	5	9
Incline(%)	Normal	0	4	10	1	10	4	8	0	3	10
Incline(%)	Pro.	3	10	4	0	8	15	7	0	5	12

Table 4.3 Walking course

Section		1	2	3	4	5	6	7	8	9	10
Time (min)		1	1	1	1	1	1	1	1	1	1
Incline(%)	Easy	0	4	0	7	0	9	0	7	0	4
Incline(%)	Normal	0	5	0	8	0	12	0	8	0	5
Incline(%)	Pro.	0	7	0	10	0	15	0	10	0	7

Table 4.4 Interval course

Section		1	2	3	4	5	6	7	8	9	10
Time (min)		1	1	1	1	1	1	1	1	1	1
Incline(%)	Easy	0	3	5	7	0	4	8	12	9	2
Incline(%)	Normal	2	8	4	8	0	4	7	10	13	5
Incline(%)	Pro.	4	7	10	0	2	10	8	3	15	7

Table 4.5 Running course

Section		WARM UP	1	2	3	4	5	6	7	COOL DOWN
Speed (km/hr)		WARM UP	3.5	4.0	4.5	5.0	4.5	4.0	3.5	COOL DOWN
Time (min)		3	3	2	2	1	2	2	3	2
Incline(%)	Easy	0	2	3	7	10	7	3	2	0
Incline(%)	Normal	0	3	5	7	10	7	5	3	0
Incline(%)	Pro.	0	4	6	8	11	8	6	4	0

Table 4.6 Cardio 20min

Section	WARM UP	1	2	3	4	5	6	7	8	9	COOL DOWN
Speed (km/hr)	WARM UP	3.5	4.0	4.5	5.0	5.0	5.0	4.5	4.0	3.5	COOL DOWN
Time (min)	3	3	2	2	2	2	2	2	2	3	2
Easy Incline(%)	0	2	0	4	0	7	0	4	0	2	0
Normal Incline(%)	0	3	1	7	1	8	1	7	1	3	0
Pro. Incline(%)	0	4	2	8	2	10	2	8	2	4	0

Table 4.7 Interval 25 min

Section	WARM UP	1	2	3	4	5	6	7	8	9	10	11	COOL DOWN
Speed (km/hr)	WARM UP	3.5	4.2	5.0	4.2	6.0	5.0	6.0	4.2	5.0	4.2	3.5	COOL DOWN
Time (min)	3	3	2	2	2	2	2	2	2	2	2	3	2
Easy Incline(%)	0	2	1	5	1	8	1	8	1	5	1	2	0
Normal Incline(%)	0	4	2	6	2	10	2	10	2	6	2	4	0
Pro. Incline (%)	0	6	4	9	4	12	4	12	4	9	4	6	0

Table 4.8 Fat Burn 30 min

Section	WARM UP	1	2	3	4	5	6	7	8	9	10	11	COOL DOWN
Speed (km/hr)	WARM UP	3.5	4.2	5.0	4.2	6.0	5.0	6.0	4.2	5.0	4.2	3.5	COOL DOWN
Time (min)	3	3	2	3	2	3	4	3	2	3	2	3	2
Easy Incline (%)	0	4	2	6	2	9	2	9	2	6	2	4	0
Normal Incline(%)	0	5	3	10	3	11	3	11	3	10	3	5	0
Pro. Incline (%)	0	7	4	10	4	12	4	12	4	10	4	7	0

Table 4.9 Endurance 35 min

