

**Application Note**

**78K0 Universal Motor  
with Chopper Control**

---

## NOTES FOR CMOS DEVICES

### ① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

Strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it once, when it has occurred. Environmental control must be adequate. When it is dry, humidifier should be used. It is recommended to avoid using insulators that easily build static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work bench and floor should be grounded. The operator should be grounded using wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with semiconductor devices on it.

### ② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

No connection for CMOS device inputs can be cause of malfunction. If no connection is provided to the input pins, it is possible that an internal input level may be generated due to noise, etc., hence causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using a pull-up or pull-down circuitry. Each unused pin should be connected to  $V_{DD}$  or GND with a resistor, if it is considered to have a possibility of being an output pin. All handling related to the unused pins must be judged device by device and related specifications governing the devices.

### ③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

Power-on does not necessarily define initial status of MOS device. Production process of MOS does not define the initial operation status of the device. Immediately after the power source is turned ON, the devices with reset function have not yet been initialized. Hence, power-on does not guarantee out-pin levels, I/O settings or contents of registers. Device is not initialized until the reset signal is received. Reset operation must be executed immediately after power-on for devices having reset function.

- The information in this document is current as of 28.04, 2004. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC sales representative for availability and additional information.
- No part of this document may be copied or reproduced in any form or by any means without prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.
- NEC Electronics does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC Electronics products listed in this document or any other liability arising from the use of such NEC Electronics products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of customer's equipment shall be done under the full responsibility of customer. NEC Electronics no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.
- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific".

The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.

- "Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots.
- "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support).
- "Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

The quality grade of NEC Electronics products is "Standard" unless otherwise expressly specified in NEC Electronics data sheets or data books, etc. If customers wish to use NEC Electronics products in applications not intended by NEC Electronics, they must contact NEC Electronics sales representative in advance to determine NEC Electronics 's willingness to support a given application.

- Notes:**
1. " NEC Electronics" as used in this statement means NEC Electronics Corporation and also includes its majority-owned subsidiaries.
  2. " NEC Electronics products" means any product developed or manufactured by or for NEC Electronics (as defined above).

M8E 02.10

# Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

## **NEC Electronics America Inc.**

Santa Clara, California  
Tel: 408-588-6000  
800-366-9782  
Fax: 408-588-6130  
800-729-9288

## **NEC Electronics (Europe) GmbH**

Duesseldorf, Germany  
Tel: 0211-65 03 1101  
Fax: 0211-65 03 1327

## **Sucursal en España**

Madrid, Spain  
Tel: 091- 504 27 87  
Fax: 091- 504 28 60

## **Succursale Française**

Vélizy-Villacoublay, France  
Tel: 01-30-67 58 00  
Fax: 01-30-67 58 99

## **Filiale Italiana**

Milano, Italy  
Tel: 02-66 75 41  
Fax: 02-66 75 42 99

## **Branch The Netherlands**

Eindhoven, The Netherlands  
Tel: 040-244 58 45  
Fax: 040-244 45 80

## **Branch Sweden**

Taeby, Sweden  
Tel: 08-63 80 820  
Fax: 08-63 80 388

## **United Kingdom Branch**

Milton Keynes, UK  
Tel: 01908-691-133  
Fax: 01908-670-290

## **NEC Electronics Hong Kong Ltd.**

Hong Kong  
Tel: 2886-9318  
Fax: 2886-9022/9044

## **NEC Electronics Hong Kong Ltd.**

Seoul Branch  
Seoul, Korea  
Tel: 02-528-0303  
Fax: 02-528-4411

## **NEC Electronics Singapore Pte. Ltd.**

Singapore  
Tel: 65-6253-8311  
Fax: 65-6250-3583

## **NEC Electronics Taiwan Ltd.**

Taipei, Taiwan  
Tel: 02-2719-2377  
Fax: 02-2719-5951

# Table of Contents

<b>Chapter 1</b>	<b>Overview</b>	<b>9</b>
1.1	Introduction	9
1.2	78K0 Family Overview	9
1.3	78K0 / KB1 Devices	11
1.4	Feature List 78K0 / KB1	12
<b>Chapter 2</b>	<b>Universal Motor</b>	<b>13</b>
2.1	Universal Motor	13
2.2	Universal Motor Speed Control	14
2.2.1	Phase Angle Control	14
2.2.2	Chopper Control	16
<b>Chapter 3</b>	<b>System Design</b>	<b>19</b>
3.1	System Concept	19
3.2	System Configuration	20
<b>Chapter 4</b>	<b>Hardware Configuration</b>	<b>21</b>
4.1	Schematic Diagram	21
4.2	Device Configuration	22
4.3	Peripherals and I/O Usage	23
4.4	16-Bit Timer Function	24
4.5	8 bit Timer / Event Counter 50	26
4.6	A/D Converter	27
<b>Chapter 5</b>	<b>Software Process Description</b>	<b>29</b>
5.1	Main	29
5.2	Micro	29
5.3	Start-up	29
5.4	PWM control	30
5.4.1	Open / Closed loop operation	30
5.5	Display	31
5.6	Memory	31
<b>Chapter 6</b>	<b>Flow Charts</b>	<b>33</b>
6.1	Main Program	33
6.2	Micro	34
6.3	Start-up	35
6.4	PWM Generation	36
6.5	Display	37
<b>Chapter 7</b>	<b>Program Listing</b>	<b>39</b>
7.1	Main	39
7.2	Micro	41
7.3	Start-up	50
7.4	PWM Generation	53
7.5	Display	58
7.6	Conversion Table	61

## List of Figures

Figure 1-1:	78K0 / KB1 Block Diagram .....	11
Figure 2-1:	Universal Motor – Operation.....	13
Figure 2-2:	Phase Angle Control Principle.....	15
Figure 2-3:	Chopper Motor Control Principle .....	17
Figure 3-1:	Chopper Control System Concept.....	19
Figure 3-2:	Chopper Control System Configuration.....	20
Figure 4-1:	Schematic Diagram .....	21
Figure 4-2:	16-bit Timer/Event Counter 00 Block Diagram .....	24
Figure 4-3:	8-bit Timer/Event Counter 50 Block Diagram .....	26
Figure 4-4:	A/D Converter Block Diagram .....	27
Figure 6-1:	Main Program .....	33
Figure 6-2:	Initialisation of the Microcontroller .....	34
Figure 6-3:	Start-up.....	35
Figure 6-4:	PWM Generation .....	36
Figure 6-5:	Display.....	37

## List of Tables

Table 1-1:	78K0 Series, Differences between Subseries .....	10
Table 4-1:	Device Configuration .....	22
Table 4-2:	I/O Configuration .....	23



# Chapter 1 Overview

## 1.1 Introduction

This application note serves as an example of a PWM control for a universal motor, using an NEC K0 8-bit microcontroller.

The aim of the application note is to provide the user with an example for driving a universal motor via an IGBT directly from the mains, using a low-cost microcontroller.

A closed-loop control is realized by measuring the motor speed with a Hall sensor.

A suitable regulation algorithm depends largely on the final application and its requirements. Therefore the one implemented in this Application Note has to be seen just as an example.

The software and hardware configurations published here are examples and are not intended for mass production.

## 1.2 78K0 Family Overview

The heart of NEC's 78K0 family is a powerful 8-bit CPU with an excellent power/performance ratio.

The 78K0 series is a highly integrated 8-bit single-chip microcontroller family.

It features CPU, ROM, RAM and peripheral functions such as A/D converter, UART, I<sup>2</sup>C and serial interfaces as well as many dedicated ASSP peripheral functions.

All devices offer a range of on-chip memory options, both as Flash and Masked ROM.

The 78K0 core provides four register banks, each with 8-bit general registers that can be concatenated to form a 16-bit register, supporting 16-bit operations.

A total of 63 instructions are available. Bit manipulation operations are supported on all registers and the entire RAM address space.

**Table 1-1: 78K0 Series, Differences between Subseries**

Function		ROM Capacity	Timer				8-bit A/D	10-bit A/D	8-bit D/A	Serial Interface	I/O	V <sub>DD</sub> MIN value	External expansion	
Subseries Name			8-bit	16-bit	WT	WDT								
Controller	μPD78075B	32K to 40K	4 ch	1 ch	1 ch	1 ch	8 ch	-	2 ch	3 ch (UART: 1 ch)	88	1.8 V	×	
	μPD78078	48K to 60K								3 ch (UART: 1 ch, I <sup>2</sup> C: 1 ch)				
	μPD78078Y									3 ch (UART: 1 ch)	61	2.7 V		
	μPD78070A	-								3 ch (UART: 1 ch, I <sup>2</sup> C: 1 ch)				
	μPD78070AY	-	1 ch	1 ch	1 ch	8 ch	-	2 ch	3 ch (time-division UART: 1 ch)	68	1.8 V			
	μPD780058	24K to 60K	3 ch (time-division UART: 1 ch)											
	μPD780058Y		3 ch (time-division UART: 1 ch, I <sup>2</sup> C: 1 ch)											
	μPD780065	40K to 48K	2 ch	2 ch	1 ch	1 ch	-	8 ch	-	4 ch (UART: 1 ch)	60	2.7 V		
	μPD780078	48K to 60K								4 ch (UART: 1 ch, I <sup>2</sup> C: 1 ch)	52	1.8 V		
	μPD780078Y									3 ch (UART: 1 ch, time-division 3-wire: 1 ch)	39			
	μPD780034AS	8K to 32K								1 ch	8 ch			51
	μPD780034A									3 ch (UART: 1 ch, I <sup>2</sup> C: 1 ch)				
μPD780034AY	8 ch									-	8 ch			
μPD780024AY	8 ch									-	8 ch			
On-chip power-on reset	78K0/KB1	8K to 24K								3 ch	1 ch	-	1 ch	-
	78K0/KC1	8K to 32K	4 ch	1 ch	1 ch	1 ch	-	8 ch	-	3 ch (UART: 2 ch)	32			
	78K0/KD1									4 ch (UART: 2 ch)	22			
	78K0/KE1	8K to 60K	2 ch	5 ch (UART: 2 ch)	22									
	78K0/KF1	24K to 60K	2 ch	5 ch (UART: 2 ch)	22	×								

### 1.3 78K0 / KB1 Devices

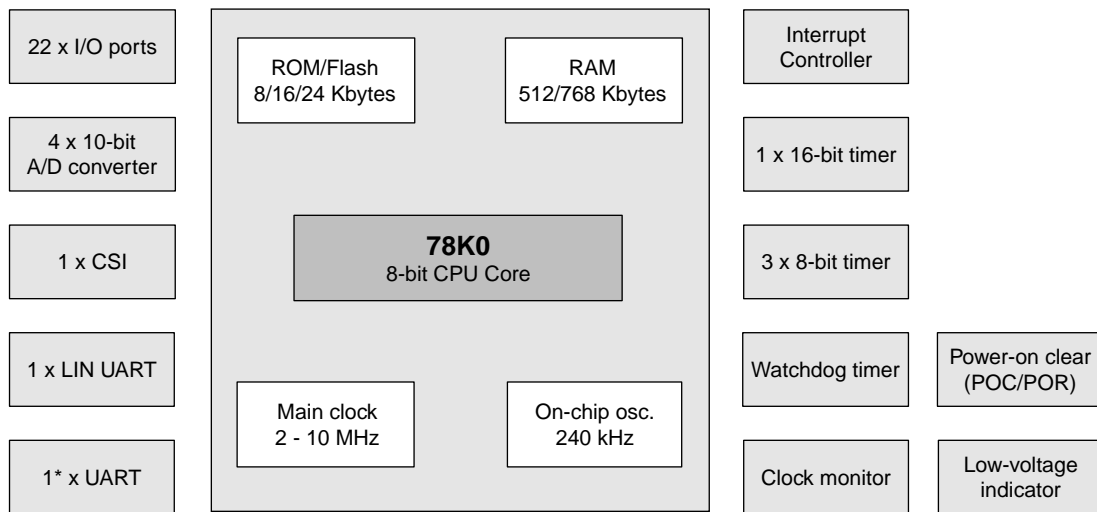
KB1 devices are part of the 78K0 microcontroller family.

With their many different integrated peripherals these devices are ideal for industrial and household appliances.

The KB1 series consists of four devices that differ mainly in RAM and ROM size.

For further details refer to the block diagram in Figure 1-1 and the feature list in section 1.4 “Feature List 78K0 / KB1” on page 12.

**Figure 1-1: 78K0 / KB1 Block Diagram**



**Note:** \* - depending of the actual device

### 1.4 Feature List 78K0 / KB1

- 8-bit CISC CPU core
- Hardwired instructions for bit manipulation, multiplication and division
- 10 MHz max. clock frequency (200 ns min. instruction cycle time)
- On-chip oscillator
- Various memory configurations (including flash memory)
- Clock monitor
- Power-on clear (POC/POR) with  $\overline{\text{RESET}}$  output
- Low-voltage indicator
- 4-channel A/D converter, 10-bit resolution
- Advanced UART (highly flexible, supports LIN by hardware)
- Standard UART (except  $\mu\text{PD780101}$ )
- 3-wire serial interface (CSI)
- $3 \times 8$ -bit timers, 16-bit timer
- Watchdog timer
- 22 I/Os
- Operating voltage: 2.7 to 5.5 V
- Operating temperature: -40 to +125°C (device dependent)
- Package: 30-pin SSOP

## Chapter 2 Universal Motor

### 2.1 Universal Motor

Universal motors are used in applications where speed control and high torque are required. Typical applications are in the field of white goods or power tools.

This motor can run on either DC or AC power, which is why it is called a universal motor.

The universal motor consists out of three main parts.

- Stator: contains the field windings to generate the radial magnetic flux
- Rotor: armature winding supplied with current via the carbon brushes to generate magnetic flux
- Brushes: mechanical linking of power supply and rotor windings

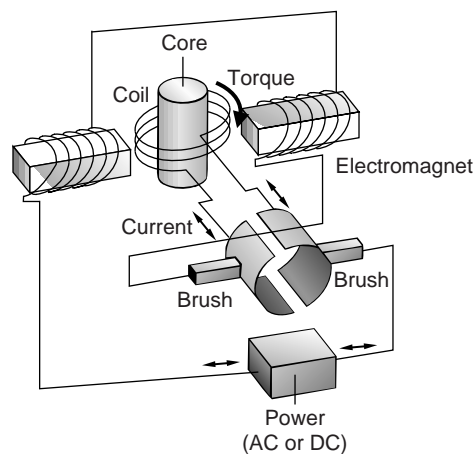
If a DC power is connected to a universal motor, the coils of the stator behave as if they were permanent magnets and the universal motor behaves like a DC motor, with one exception: it does not reverse direction when the current passing through it is reversed. It will continue to rotate in the same direction, because reversing the current through the rotor also reverses the current through the electromagnets. In effect, every pole of the motor changes from north to south or from south to north. As all the poles change their polarity, the motor will continue turning in the same direction.

As the universal motor always rotates in the same direction irrespective of the direction of the current, it works also when connected to AC power.

In order to change the motor's direction of rotation, the coils of the stator must be rewired, because this reverses the poles and thus the direction of rotation.

When running a universal motor on AC power, the motor experiences no torque during current reversal, but the average torque is sufficient to turn the motor as if it were connected to a DC power supply.

**Figure 2-1: Universal Motor – Operation**



### 2.2 Universal Motor Speed Control

One major requirement for electromotors used in domestic appliances is the ability to control its speed.

Two electronic solutions, phase angle control and chopper control, are widely used in industry today to realize speed control.

Both solutions have their strengths and weaknesses.

#### 2.2.1 Phase Angle Control

Phase angle control is a simple and cost-effective solution to change the speed of a universal motor by using a triac.

The gate of the triac can be controlled directly by the output ports of a microcontroller and the whole circuitry is usually connected directly to the mains.

The advantages of this solution are:

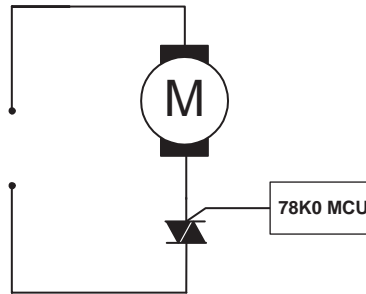
- simplicity
- cost effectiveness
- full speed control

On the other hand, the drawbacks are:

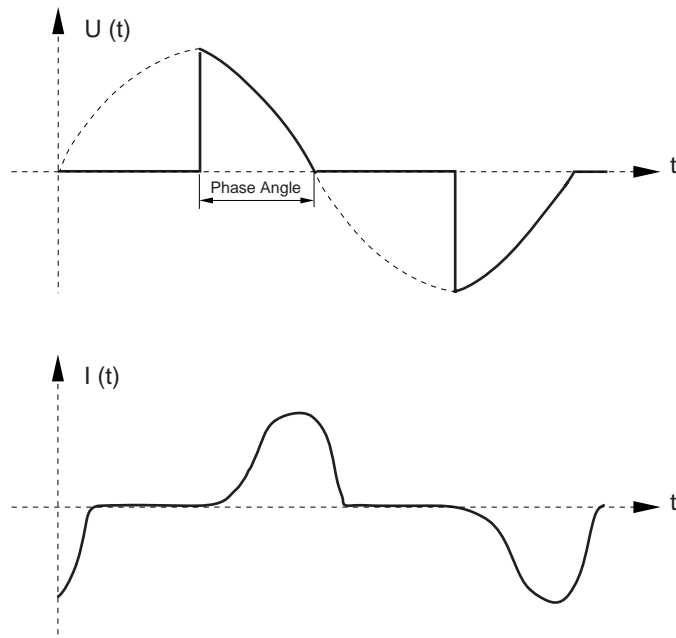
- high brush noise
- short lifetime
- high current ripple
- poor EMI behaviour

Figure 2-2: Phase Angle Control Principle

(a) Block Diagram



(b) Timing



### 2.2.2 Chopper Control

PWM control is another solution for controlling the speed of a universal motor.

The AC voltage from the mains is rectified and then switched at a high frequency by a Power MOSFET or an IGBT.

The switching frequency is usually in the range of 10 to 20 KHz.

PWM control is the more advanced solution for controlling the speed of a universal motor. Driving a universal motor in PWM mode results in lower current ripple, which reduces the harmonics and therefore results in much better EMI behaviour.

The reduction of harmonics compared with phase angle control also reduces the copper and iron losses, which results in higher motor efficiency.

Running a motor in chopper mode also reduces acoustic noise, which is an important factor nowadays that should not be underestimated.

The advantages of this solution are:

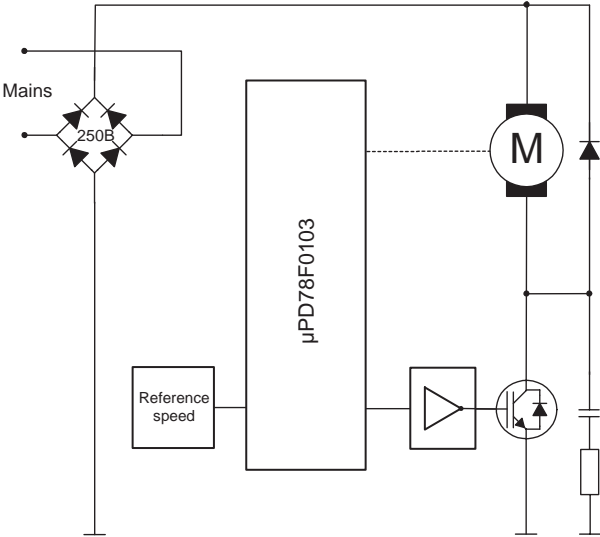
- higher motor efficiency
- lower acoustic noise
- smooth current behaviour

On the other hand, the drawbacks are:

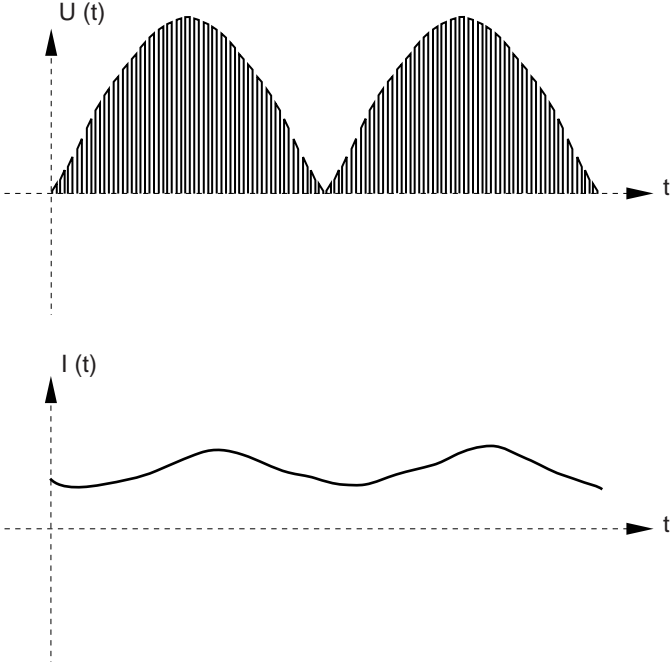
- more components needed
- price disadvantage
- filtering of chopper frequency required

Figure 2-3: Chopper Motor Control Principle

(a) Block Diagram



(b) Timing



**[MEMO]**

## Chapter 3 System Design

### 3.1 System Concept

The aim of this application note is to provide a speed regulation example realized by chopper control, using an 8-bit microcontroller from NEC's 78K0 family.

The  $\mu$ PD78F0103 was selected for this application example.

The universal motor runs on rectified main voltage that is PWM-modulated.

An IGBT switches the rectified mains voltage via a suitable driver.

The PWM is generated by NEC's 8-bit microcontroller.

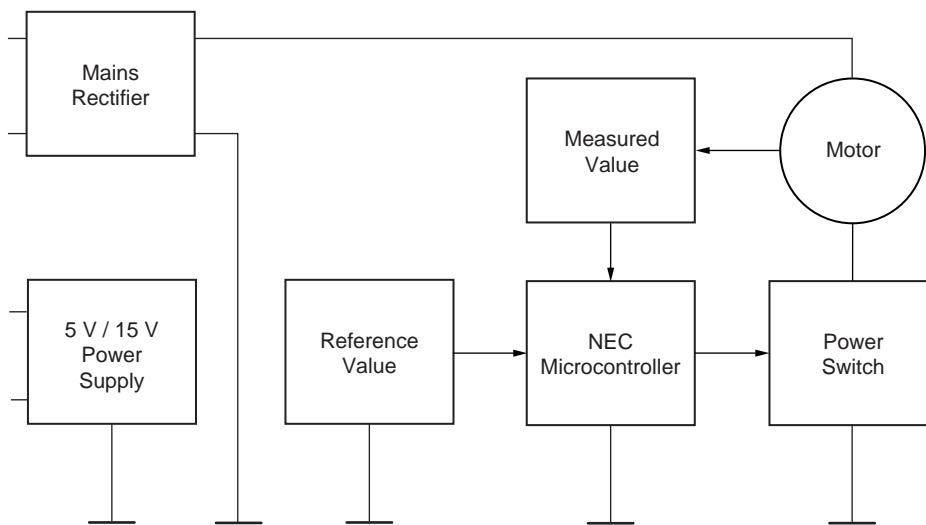
The motor speed can be controlled by changing the duty cycle of the generated PWM signal.

The motor speed is measured via a Hall sensor. A potentiometer, connected to the A/D converter of the microcontroller, delivers the reference value for the speed control function.

A closed-loop system is realized by comparing the reference and measured value.

Depending on the calculation results, the duty cycle of the PWM signal, controlling the power switch, is adapted to achieve a stable motor speed.

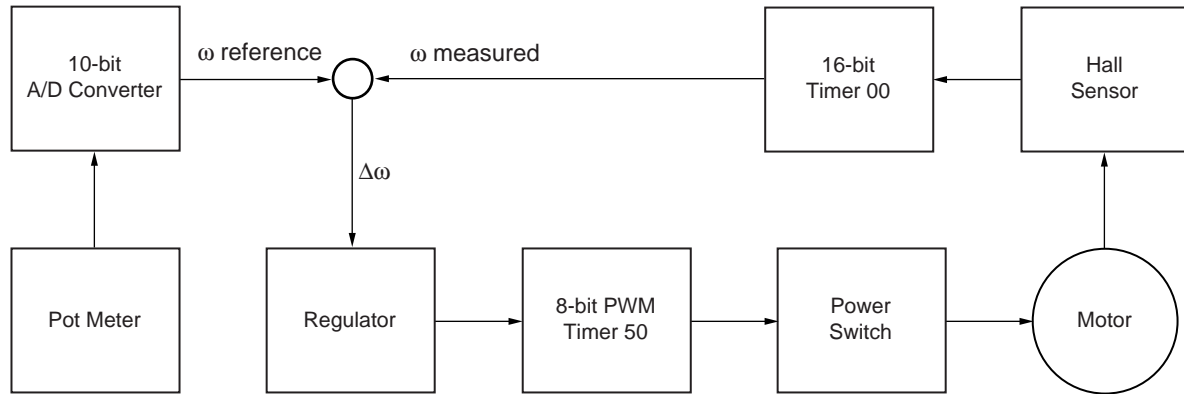
**Figure 3-1: Chopper Control System Concept**



### 3.2 System Configuration

The system configuration and the relationship between control software and hardware of the chopper controlled Universal Motor is shown in detail in Figure 3-2.

**Figure 3-2: Chopper Control System Configuration**



The motor speed information is given by the output signal of the Hall sensor. This signal is measured by the 16-bit timer/counter, operating in pulse-width measuring mode.

The reference value for the closed-loop control system is given by an analog potentiometer, connected to the on-chip A/D converter of the microcontroller.

The speed reference value is compared with measured value to generate the input signal to the regulator.

The output signal of the regulator itself changes the duty cycle of the 8-bit PWM counter in a way that constant motor speed is achieved.

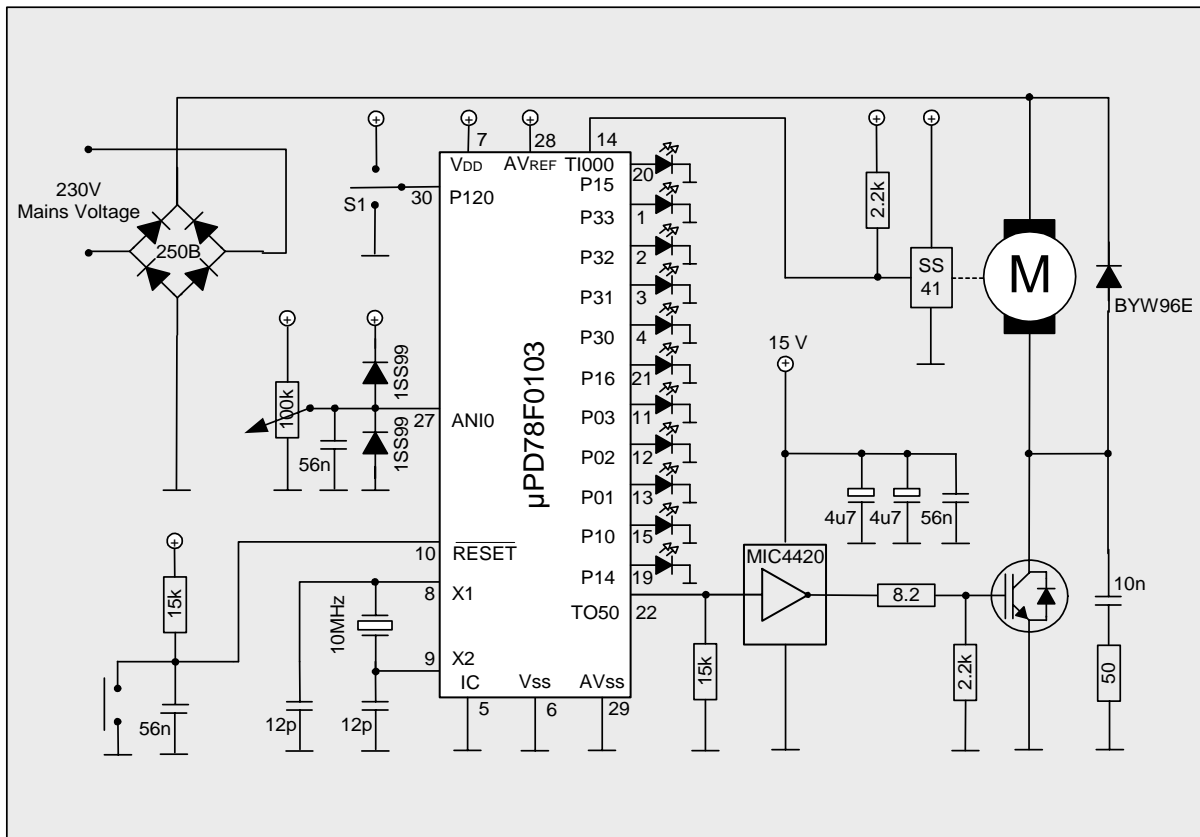
The function of each of the peripheral units is described in the next chapter.

## Chapter 4 Hardware Configuration

### 4.1 Schematic Diagram

The diagram shows the chopper motor control design using a  $\mu$ PD78F0103 microcontroller.

Figure 4-1: Schematic Diagram



Switch S1 allows the selection between closed-loop and open-loop system operation.

## 4.2 Device Configuration

The configuration and operating environment of the  $\mu$ PD78F0103 is shown in Table 4-1.

**Table 4-1: Device Configuration**

Operating clock	10.00 MHz
Operating voltage	5 V
Internal ROM	24 KB
Internal RAM	768 Bytes

### 4.3 Peripherals and I/O Usage

Table 4-2 shows the microcontroller's I/O pins. The pins used in the application and their functions are listed.

**Table 4-2: I/O Configuration**

Pin No.:	Pin Name	Signal Name	Function
1	P33		LED
2	P32		LED
3	P31		LED
4	P30		LED
5	IC	V <sub>PP</sub>	Programming
6	V <sub>SS</sub>		Ground
7	V <sub>DD</sub>		V <sub>DD</sub>
8	X1		Main crystal
9	X2		Main crystal
10	RESET		System reset
11	P03		LED
12	P02		LED
13	P01		LED
14	P00	TI000	Speed measure
15	P10		LED
16	P11		Not used
17	P12		Not used
18	P13		Not used
19	P14		LED
20	P15		LED
21	P16		LED
22	P17	TO50	PWM out
23	P130		Debugging
24	P23		Not used
25	P22		Not used
26	P21		Not used
27	P20	ANI0	Speed reference
28	AV <sub>REF</sub>	5 V	A/D reference
29	AV <sub>SS</sub>	0 V	Analog ground
30	P120	I/O	Switch

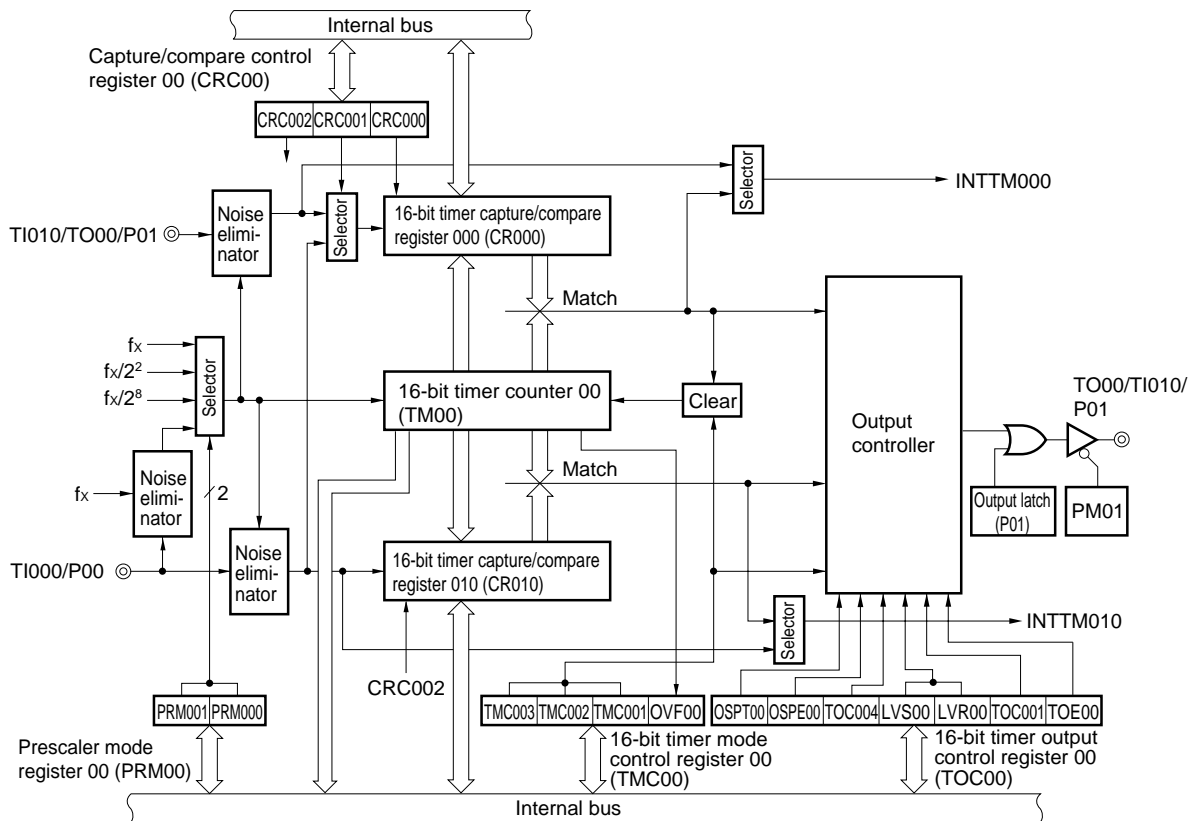
### 4.4 16-Bit Timer Function

As shown in Figure 3-2, “Chopper Control System Configuration,” on page 20, the 16-bit timer is used to measure the motor speed.

The 16-bit timer (TM00) of the  $\mu$ PD78K0103 has several operating modes:

- Interval timer
  - Generates interrupt request at the preset time interval
- PPG mode
  - Outputs a square wave, output pulse and frequency can be set
- Pulse width measurement
  - Measures pulse width of an external signal
- External event counter
  - Measures the number of external pulses
- Square wave output
  - Generates a square wave signal with an programmable frequency

**Figure 4-2: 16-bit Timer/Event Counter 00 Block Diagram**



The 16-bit timer is set to pulse width measurement mode to measure the frequency of the square wave signal.

A magnetic ring with eight pole pairs is mounted on the axis of the motor rotor.

The Hall sensor, mounted close to the magnetic ring, generates a square-wave signal.

The generated frequency of the Hall sensor is in the range of 300 Hz at low speed and increases to a maximum speed of 1800 Hz.

The count clock frequency of the 16-bit timer is programmable. In this application example, a frequency of 39 KHz is selected to measure the pulse width of the Hall sensor signal.

The pulse-width measurement is performed in a so-called "by means of restart" mode.

The speed measurement in this application example is not interrupt-controlled but performed in free-running mode.

The 16-bit timer is triggered by the rising edge of the Hall sensor signal.

A rising edge transfers the value of the 16-bit timer/counter to the CR010 capture/compare register and automatically clears and restarts TM00 for the next measurement.

This has the advantage that the last measured speed value is always available in CR010 and can be used at any time for the speed control algorithm.

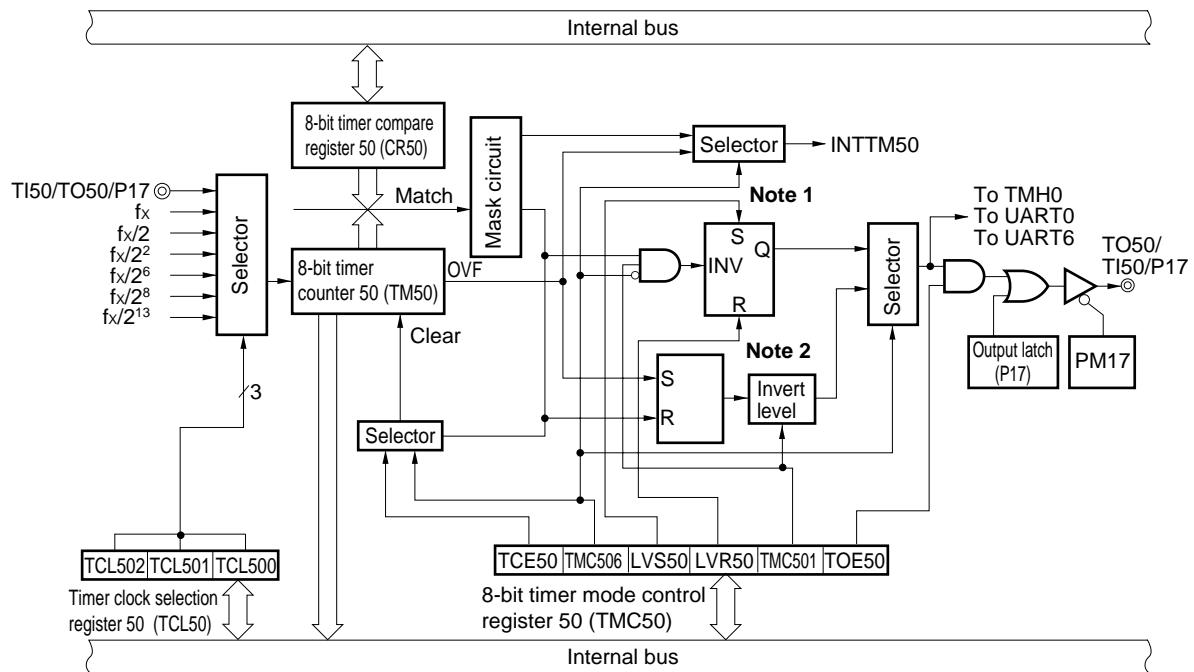
### 4.5 8 bit Timer / Event Counter 50

This timer is used to generate a PWM signal that controls the speed of the motor via the IGBT driver and IGBT. The PWM signal is varied by modifying the duty cycle.

The timer provides the following functions:

- interval timer
- external event counter
- square-wave output
- PWM output

**Figure 4-3: 8-bit Timer/Event Counter 50 Block Diagram**



**Notes: 1.** Timer output F/F

**2.** PWM output F/F

PWM output mode is selected and an output frequency of 10 KHz. The duty ratio of the 10 KHz signal is determined by the value of the 8-bit timer compare register 50 (CR50).

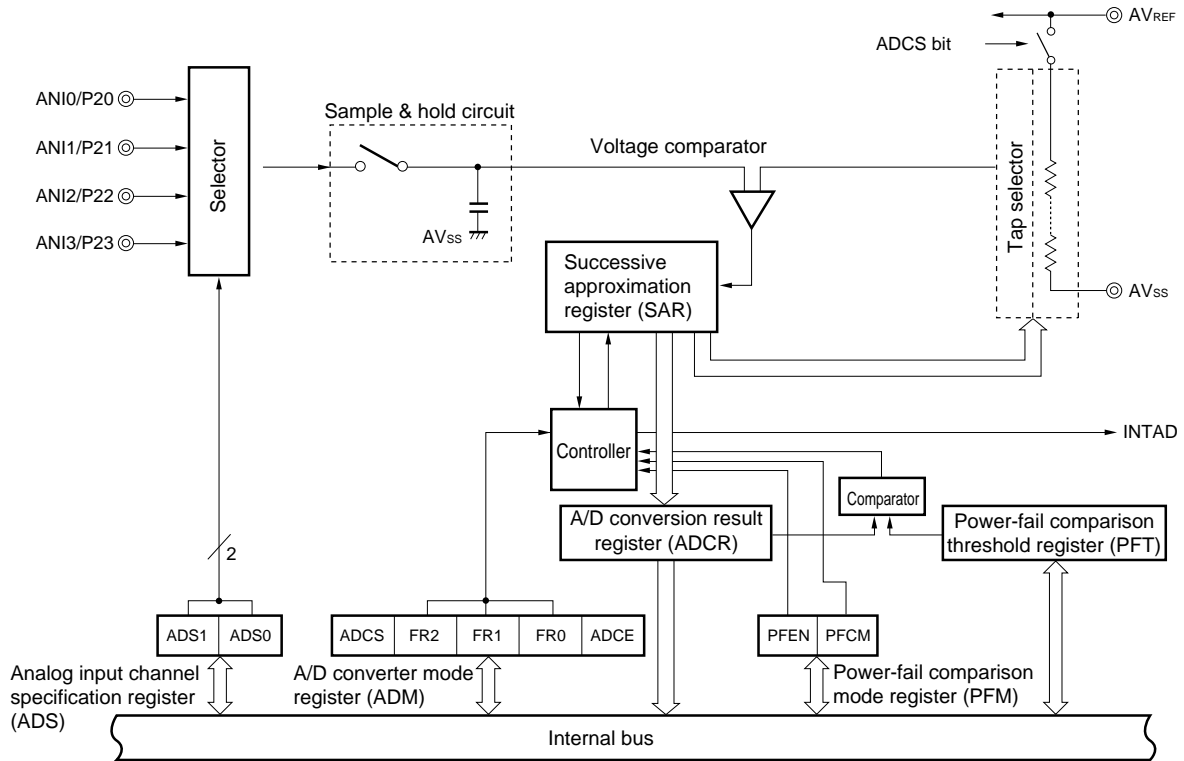
The calculation result of the speed control algorithm can be set on the fly, i.e., the CR50 register can be set at any time without stopping the PWM mode of the 8-bit timer.

A subroutine structure is used to modify the duty cycle by rewriting the CR50 register.

## 4.6 A/D Converter

The on-chip A/D converter has 4 input channels and a 10-bit resolution.

Figure 4-4: A/D Converter Block Diagram



In the example, channel one (ANI0) is connected to a potentiometer connected between  $V_{DD}$  and ground.

The A/D converted potentiometer value represents the reference value for the speed control algorithm for regulating the speed in the closed-loop system of this example.

The A/D conversion process is continuous and always the last converted value can be read out from the ADCR register of the  $\mu$ PD78F0103.

The conversion time of the  $\mu$ PD78F0103 is 28.8  $\mu$ s.

Analog reference  $AV_{DD}$  and analog ground of the A/D converter are connected respectively to  $V_{DD}$  and  $V_{SS}$ .

[MEMO]

## Chapter 5 Software Process Description

### 5.1 Main

The main program provides the framework within which the subroutines execute the various tasks of the closed-loop motor control application example.

The main program starts by initializing the stack pointer and device-specific IMS register settings, before invoking the global initialization subroutine called Micro.

The start-up routine performs a soft start-up of the universal motor until the reference speed is reached.

The endless loop of the main program consists of the PWM generation and the Display subroutine.

### 5.2 Micro

The Micro subroutine is responsible for initializing the system after a system reset.

It configures the basic clock settings of the device, selects the oscillators running the device during the different operating modes and initializes the used peripherals.

It has the following sections:

- Watchdog timer
- Initialization and switching of CPU clock
- Initialization of I/O ports
- A/D converter settings (reference value generation)
- PWM timer initialization (PWM generation)
- Timer 00 setting (speed measurement)

### 5.3 Start-up

This routine performs a soft start-up of the universal motor.

The A/D converter delivers the reference value of the motor speed.

After determining this value, the PWM register of Timer 50 is set to minimum speed.

Timer 50 generates the PWM signal for controlling the motor speed.

PWM generation of Timer 50 is enabled and the PWM register, containing the minimum speed value, is incremented to increase the motor speed until the reference speed is reached.

To achieve a soft start-up, a delay is executed after each change in the PWM duty cycle.

### 5.4 PWM control

This subroutine controls the motor speed.

In this application, controlling the motor speed means comparing reference and measured speed and calculating the speed error from the difference.

Speed error in this context means the motor is either running too fast, too slow or runs exactly at reference speed, in which case the speed error is zero.

Depending on the calculated speed error, the duty cycle of the generated PWM signal is modified by rewriting the CR50 register of Timer 50.

Modifying the PWM signal basically means incrementing or decrementing the corresponding variable (PWM\_reference) in dependency on the calculated speed error.

The reference speed in this application is given by a potentiometer.

The A/D converter continuously converts its input value, i.e., approx. every 30  $\mu$ s a new reference speed value is provided to the ADCR register for further calculations.

The A/D converter has a 10-bit resolution, which in this application is standardized to 8 bits.

The motor speed is measured with Timer 00 of the  $\mu$ PD78K0103.

Like the A/D converter, the speed measurement resolution is also standardized to 8 bits.

The timer runs in a so-called "pulse width measurement by means of restart" mode, i.e., the timer clears and starts its counter function automatically at the specified edges and provides the counter value for further calculation in its CR000 register.

A magnetic ring with eight pole pairs is mounted on the rotor of the universal motor.

A Hall sensor, mounted close to the ring provides speed information to Timer 00.

While calculating a new PWM duty cycle it is very important to check whether any underflow or overflow of the system limit values has been occurred.

'Regulation delay' is a function implemented in the PWM control subroutine. This is necessary to optimize the control performance of the whole system.

By changing the constants 'regulation\_delay\_const\_out' and 'regulation\_delay\_const\_in' this scheme can easily be adapted for transfer to other systems.

#### 5.4.1 Open / Closed loop operation

This motor control application can run either in closed-loop or in open-loop mode.

The modes are selected via a switch connected to port P120.

In closed-loop operation the regulation algorithm as described above is performed.

In an open-loop system, this algorithm is switched off and basically the reference value provided in the ADCR register is transferred directly to the CR50 register of Timer 50, which specifies the duty cycle and thus the motor speed.

To achieve similar speeds in closed- and open-loop operation requires a conversion table with reference values.

The reason for that is in the different control characteristic curves in open- and closed-loop operation within this application.

### 5.5 Display

Display is a subroutine that in closed-loop operation indicates the regulation accuracy. In open-loop mode the display is switched off.

The accuracy of the regulation is calculated from the reference and measured speed values.

11 LEDs indicate the regulation accuracy in 2% steps.

The green LED on port P16 indicates 100% accuracy. The red LEDs on ports P30, P31, P32, P33 and P01, P02, P03, P10 indicate the accuracy in 2% steps in positive and negative direction, respectively.

The yellow LEDs on port P15 and P14 indicate an accuracy of less than 8% in positive and negative direction, respectively.

### 5.6 Memory

These application example provides an approach to a basic chopper control system suitable for implementation with NEC's 78K0 8-bit Microcontroller family.

The memory requirements for these example are as follows:

- 78K0 –  $\mu$ PD78F0103

CODE memory	16 Kbytes available,	679 bytes used
DATA memory	768 bytes available	21 bytes used

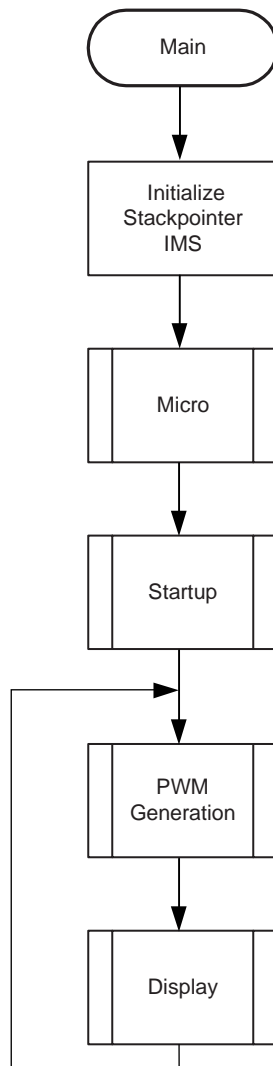
The 78K0 Microcontroller used for this application example provides further resources for additional user functions or/and adaptations to the regulation algorithm.

[MEMO]

## Chapter 6 Flow Charts

### 6.1 Main Program

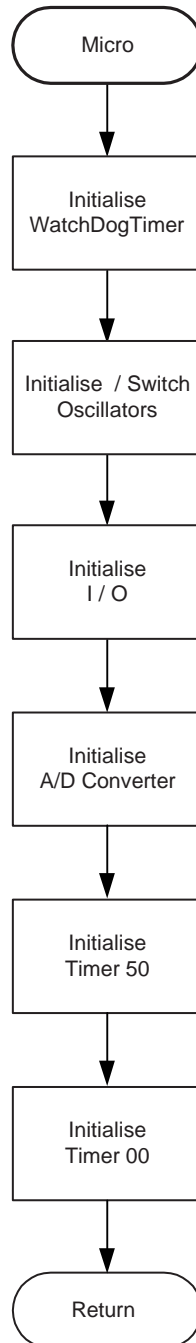
**Figure 6-1: Main Program**



## 6.2 Micro

This Module executes the initialisation of the microcontroller

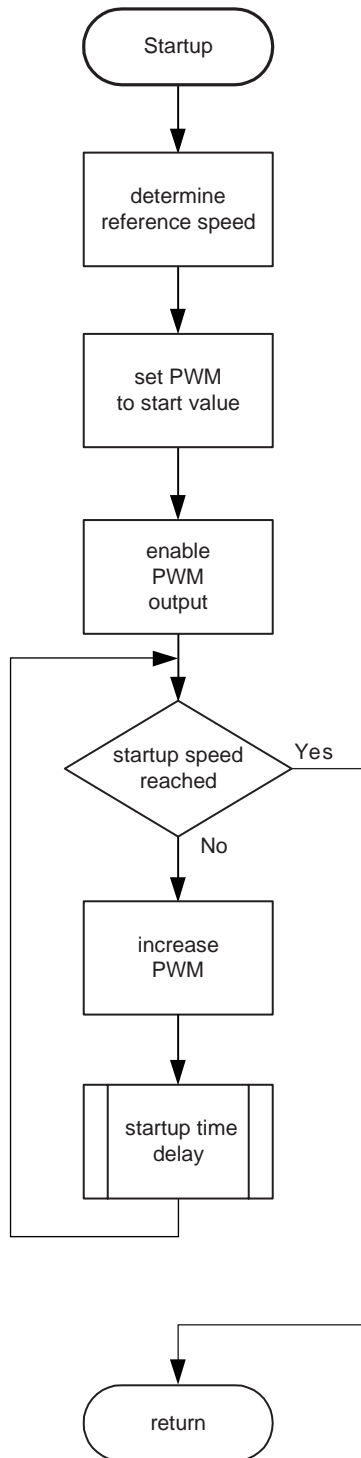
**Figure 6-2: Initialisation of the Microcontroller**



### 6.3 Start-up

This module performs soft start-up up to reference speed.

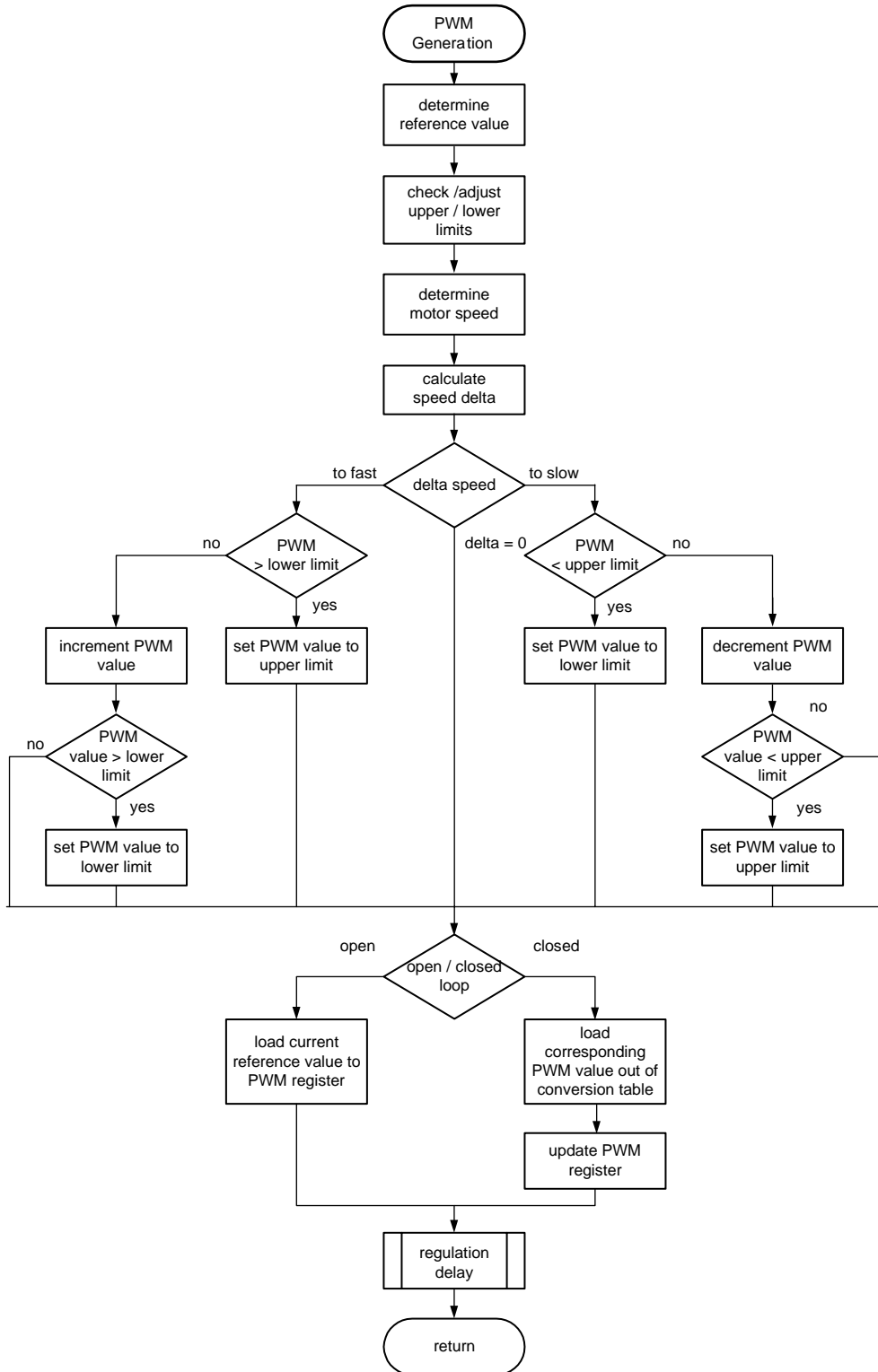
Figure 6-3: Start-up



### 6.4 PWM Generation

This module calculates out of reference and measured speed the new duty cycle of the PWM signal driving the Universal Motor.

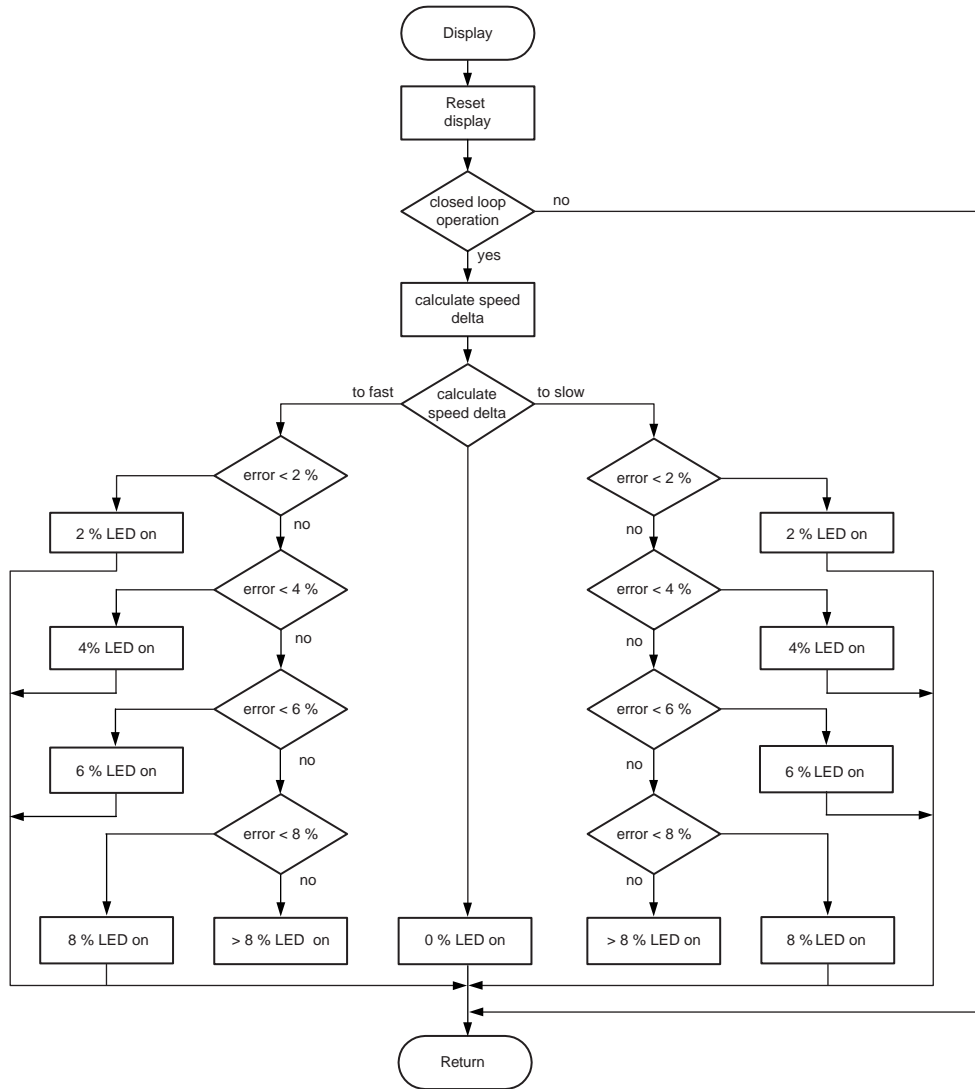
Figure 6-4: PWM Generation



### 6.5 Display

This module gives in 2% steps an indication on the achieved accuracy while running the application in closed loop mode.

Figure 6-5: Display



**[MEMO]**

## Chapter 7 Program Listing

### 7.1 Main

```

/*****
/*
/* File:          main.asm
/* Project:       Motor Control ( chopper ) - startup
/*
/* Description:   Contains power-on reset initialisation and the idle loop.*/
/*
/* History:       031103 creation
/*
*****/
; Enviroment   Device:      uPD78F0103
;               Assembler:  A78000      V3.21A/386
;               Compiler:   ICC78000    V3.21A/386
;               Linker:     XLINK        V4.51J/386
;=====
;Changes
;=====

    NAME      Main          ; module name
;   P780103    ; target device

#include      "df0103.h"    ; header file

;=====
; EXPORT LIST
;=====
; Constants:
; Variables:
; Flags:
; Labels:
PUBLIC Start              ; reset entry address

;=====
; IMPORT LIST
;=====
; Constants:
; Variables:
; Flags:
; Labels:
EXTERN InitMicro         ; from Micro
EXTERN Startup           ; from Startup
EXTERN loop              ; from PWM_generation
EXTERN Display           ; from Display

;-----

    org      0000H
    dw      Start
```

## Chapter 7 Program Listing

---

```
=====
; Segment name:  CODE
; Segment type:  relocatable code segment
; Address area:  internal ROM area
=====
RSEG          CODE
=====

;=====
; Block type:    Program idle loop
; Purpose:       Initilisation
; Input:         none
; Output:        none
;=====

Start:         movw   AX,#0FE20h           ; set stackpointer
               movw   SP,AX
               mov    IMS,#06H
               di                    ; disable Interrupts
               call   InitMicro          ; switching ring -> main oscillator
               call   Startup

EndLess:       nop                       ; endless loop
               call   loop
               call   Display
               nop
               br     EndLess

               END
```

## 7.2 Micro

```

/*****
/*
/* File:          micro.asm
/* Project:       Motor Control ( chopper ) - startup
/*
/* Description:  Initilization of CPU and peripherals
/*
/* History:       031103  creation
/*
*****/
; Enviroment      Device:      uPD78F0103
;                  Assembler:  A78000      V3.21A/386
;                  Compiler:    ICC78000    V3.21A/386
;                  Linker:      XLINK       V4.51J/386
;=====
;Changes
;=====

NAME    micro          ; module name
;P780103          ; target device

#include"df0103.h"      ; header file

;=====
; EXPORT LIST
;=====
; Constants:
; Variables:
; Flags:
; Labels:
PUBLIC  InitMicro      ; - module initilization

;=====
;IMPORT LIST
;=====
; Constants:
; Variables:
; Flags:
; Labels:

;=====
; Segment name:  CODE
; Segment type:  relocatable code segment
; Address area:  internal ROM area
;=====
RSEG          CODE
;=====

;=====
; Block type:    global standard subroutine
; Purpose:       Initilization for chopper mode motor control
; Input:         none
; Output:        none
;=====

```

## Chapter 7 Program Listing

---

```

InitMicro:  call    InitWDTM           ; Begin
            call    InitOsc           ;
            call    InitIO            ;
            call    InitAD            ;
            call    InitTM50          ;
            call    InitTM00          ;
            ret      ; End

;=====
; Block type:      local standard subroutine
; Purpose:         Initialization / Switching of CPU clock
; Input:           none
; Output:          none
;=====

cPCCini     equ     00000000B         ; processor clock control register
            ; | | | | | | | |
            ; | | | | | | | |           PCCn   fCPU
            ; | | | | | | | |           -----
            ; | | | | | | | |           000    fx
            ; | | | | | | | |           001    fx/2
            ; | | | | | | | |           010    fx/4
            ; | | | | | | | |           011    fx/8
            ; | | | | | | | |           100    fx/16
            ; | | | | | | | |           other settings prohibited
            ; | | | | | | | |
            ; | | | | | | | |           fixed to 0
            ; | | | | | | | |           fixed to 0
            ; | | | | | | | |           fixed to 0
            ; | | | | | | | |           fixed to 0
            ; | | | | | | | |           fixed to 0

cRCMini     equ     00000000B         ; Ring-Osc mode register
            ; | | | | | | | |
            ; | | | | | | | |           RSTOP
            ; | | | | | | | |           -----
            ; | | | | | | | |           0      Ring-OSC working
            ; | | | | | | | |           1      Ring-OSC stopped
            ; | | | | | | | |
            ; | | | | | | | |           fixed to 0
            ; | | | | | | | |           fixed to 0
            ; | | | | | | | |           fixed to 0
            ; | | | | | | | |           fixed to 0
            ; | | | | | | | |           fixed to 0
            ; | | | | | | | |           fixed to 0

```

## Chapter 7 Program Listing

```

cMCMMini    equ    00000000B           ; Main Clock Mode Register
; | | | | | | | |
; | | | | | | | | +----- MCM0, [0] Ring-OSC input,[1] x1 input
; | | | | | | | | +----- MCS, [0] Ring-OSC or [1] x1 working
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0

cMOCcini    equ    00000000B           ; Main Oscillator Control Register
; | | | | | | | |
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- MSTOP, [0] x1 osc, [1] x1 stopped

cOSTCini    equ    00000000B           ; Osc. Stab. Time Counter status reg.
; | | | | | | | |
; | | | | | | | |
; | | | | | | | |
; | | | | | | | | +----- MOSTn   t
; | | | | | | | | +----- -----
; | | | | | | | | +----- 10000   204.8us
; | | | | | | | | +----- 11000   819.2us
; | | | | | | | | +----- 11100   1.64ms
; | | | | | | | | +----- 11110   3.27ms
; | | | | | | | | +----- 11111   6.55ms
; | | | | | | | |
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0

cOSTSini    equ    00000101B           ; Osc. Stab. Time Select register
; | | | | | | | |
; | | | | | | | |
; | | | | | | | |
; | | | | | | | | +----- OSTSn   t
; | | | | | | | | +----- -----
; | | | | | | | | +----- 001     204.8us
; | | | | | | | | +----- 010     819.2us
; | | | | | | | | +----- 011     1.64ms
; | | | | | | | | +----- 100     3.27ms
; | | | | | | | | +----- 101     6.55ms
; | | | | | | | |
; | | | | | | | |
; | | | | | | | |
; | | | | | | | | +----- other settings prohibited
; | | | | | | | |
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0
; | | | | | | | | +----- fixed to 0

```

## Chapter 7 Program Listing

```

InitOsc:  mov     PCC,#cPCCini  ; fx=10MHz selectiert
          mov     RCM,#cRCMini ; Ring-OSC working
          mov     MCM,#cMCMini ; Ring-OSC supplied to CPU
          mov     MOC,#cMOCini ; x1 working
          mov     OSTs,#cOSTSini; 6,5ms stabilization time selected

wait:    BF     OSTC.4,wait   ; check stabilization time
          setl   MCM.0        ; after stabilization time switch to x1

          ret                    ;

```

```

;=====
; Block type:      local standard subroutine
; Purpose:         Initilization Watchdogtimer
; Input:           none
; Output:          none
;=====

```

```

cWDTMini equ     01111111B          ; Watchdog Timer Mode register
          ; | | | | | | | |
          ; | | | | | | | |
          ; | | | | | | | |
          ; | | | | |-----|-----|-----|-----|-----|-----|-----|-----|
          ; | | | | | | | |          WDCSn   Ring-OSC   x1 OSC
          ; | | | | | | | |          -----|-----|-----|-----|-----|-----|-----|-----|
          ; | | | | | | | |          000     8.53ms     819.2us
          ; | | | | | | | |          001     17.07ms    1.64ms
          ; | | | | | | | |          010     34.13ms    3.28ms
          ; | | | | | | | |          011     68.27ms    6.55ms
          ; | | | | | | | |          100    136.53ms   13.11ms
          ; | | | | | | | |          101    273.07ms   26.21ms
          ; | | | | | | | |          110    546.13ms   52.43ms
          ; | | | | | | | |          111     1.09s     104.86ms
          ; | | | | | | | |
          ; | | | | | | | |          WDCSn   clock selection
          ; | | | | | | | |          -----|-----|-----|-----|-----|-----|-----|-----|
          ; | | | | | | | |          00     fr Ring-OSC clock
          ; | | | | | | | |          01     fx Main Oscillator
          ; | | | | | | | |          1x     Watchdog timer stopped
          ; | | | | | | | |
          ; | | | | | | | |          +-----|-----|-----|-----|-----|-----|-----|-----|
          ; | | | | | | | |          +-----|-----|-----|-----|-----|-----|-----|-----|
          ; | | | | | | | |          +-----|-----|-----|-----|-----|-----|-----|-----|
          ; | | | | | | | |          fixed to 1
          ; | | | | | | | |          fixed to 1
          ; | | | | | | | |          fixed to 0

```

```

InitWDTM mov     WDTM,#cWDTMini    ; Watchdog timer stopped
          mov     IMS,#06H
          ret

```



## Chapter 7 Program Listing

```

;=====
; Block type:      local standard subroutine
; Purpose:        Initilization of A/D Converter
; Input:          none
; Output:         none
;=====

cADMini    equ      10000001B      ; A/D Converter Mode Register
; | | | | | | | |
; | | | | | | | | +----- ADcS, [0] stops conversion,
; | | | | | | | | +----- [1] enables conversion
; | | | | | | | | +----- unused, fixed to 0
; | | | | | | | | +----- unused, fixed to 0
; | | | | | | | |
; | | | | | | | |          FRn   Conversion time
; | | | | | | | |          -----
; | | | | | | | | +----- 000   288/fx
; | | | | | | | |          001   240/fx
; | | | | | | | |          010   192/fx
; | | | | | | | |          010   144/fx
; | | | | | | | |          101   120/fx
; | | | | | | | |          110    96/fx
; | | | | | | | | +----- unused, fixed to 0
; | | | | | | | | +----- ADCE,ref.volt.gen.,[0] off,[1] on

cADSini    equ      00000000B ; Analog Input Channel Specification Register
; | | | | | | | |
; | | | | | | | |          ADSn   input channel
; | | | | | | | |          -----
; | | | | | | | | +----- 000   ANI0
; | | | | | | | |          001   ANI1
; | | | | | | | |          010   ANI2
; | | | | | | | |          011   ANI3
; | | | | | | | |
; | | | | | | | |
; | | | | | | | | +----- unused, fixed to 0
; | | | | | | | | +----- unused, fixed to 0
; | | | | | | | | +----- unused, fixed to 0
; | | | | | | | | +----- unused, fixed to 0
; | | | | | | | | +----- unused, fixed to 0

InitAD:    mov      ADM,#cADMini      ; start A/D Converter
           mov      ADS,#cADSini     ; select analog input channel ANI0

           ret

```

## Chapter 7 Program Listing

```

;=====
; Block type:      local standard subroutine
; Purpose:        Initilization of Timer 50 ( PWM out )
; Input:          none
; Output:         none
;=====

cTCL50ini  equ    00000100B          ; Timer Clock Selection Register 50
; | | | | | | | | | |
; | | | | | | | | | |          TCL50n  count clock selection
; | | | | | | | | | |
; | | | | | | | | | |          000    TI50 fallinge edge
; | | | | | | | | | |          001    TI50 rising edge
; | | | | | | | | | |          010    10      MHz (fx)
; | | | | | | | | | |          011    5       MHz
; | | | | | | | | | |          100    2.5    MHz
; | | | | | | | | | |          101    156.25 kHz
; | | | | | | | | | |          110    39.06  kHz
; | | | | | | | | | |          111    1.22   kHz
; | | | | | | | | | |
; | | | | | | | | | |          unused, fixed to 0
; | | | | | | | | | |          unused, fixed to 0
; | | | | | | | | | |          unused, fixed to 0
; | | | | | | | | | |          unused, fixed to 0
; | | | | | | | | | |          unused, fixed to 0

cTMC50ini  equ    01000010B          ; format Timer Mode Control register
; | | | | | | | | | |
; | | | | | | | | | |          TCE50, counter [0]stopped, [1]starts
; | | | | | | | | | |          TMC506, [0]clear & start, [1]PWM mode
; | | | | | | | | | |
; | | | | | | | | | |          LVS50  LVR50  Timer output F/F status
; | | | | | | | | | |
; | | | | | | | | | |          0      0      no change
; | | | | | | | | | |          0      1      output F/F reset (0)
; | | | | | | | | | |          1      0      output F/F set (1)
; | | | | | | | | | |          1      1      Setting prohibited
; | | | | | | | | | |
; | | | | | | | | | |          unused, fixed to 0
; | | | | | | | | | |          unused, fixed to 0
; | | | | | | | | | |          TMC501, active,[0]high,[1]low
; | | | | | | | | | |          TOE50, output,[0]disabled,[1]enabled

;-----

InitTM50:  mov     TCL50,#cTCL50ini    ; count clock selection
           mov     TMC50,#cTMC50ini   ; select PWM mode, output enable

           ret

```

## Chapter 7 Program Listing

```

;=====
; Block type:      local standard subroutine
; Purpose:        Initilization of 16bit Timer 00 ( motor speed counter )
; Input:          none
; Output:         none
;=====
cTMC00ini   equ    00001010B           ; 16bit Timer Mode Control Register
; | | | | | | | |
; | | | | | | | +-----; OVF00, Overflow detected,[0]no, [1]yes
; | | | | | | |
; | | | | | | |           timer operating modes
; | | | | | | | -----
; | | | | | | | +----- 000   operating stop
; | | | | | | |           001   operating stop
; | | | | | | |           010   free running mode
; | | | | | | |           011   free running mode
; | | | | | | |           100   clear and start valid edge
; | | | | | | |           101   clear and start valid edge
; | | | | | | |           110   clear and start match TM/CR
; | | | | | | |           111   clear and start match TM/CR
; | | | | | | |
; | | | | | | | +----- unused, fixed to 0
; | | | | | | | +----- unused, fixed to 0
; | | | | | | | +----- unused, fixed to 0
; | | | | | | | +----- unused, fixed to 0

cCRC00ini   equ    00000111B           ; Capture/Compare Control Register
; | | | | | | | |
; | | | | | | |           operating mode selection
; | | | | | | | +----- CRC000, operates [0]compare [1]capture
; | | | | | | |
; | | | | | | |           capture trigger selection
; | | | | | | | +----- CRC000,capture [0]valid[1]reverse edge
; | | | | | | |
; | | | | | | |           operating mode selection
; | | | | | | | +----- CRC010, operates [0]compare [1]capture
; | | | | | | | +----- unused, fixed to 0
; | | | | | | | +----- unused, fixed to 0
; | | | | | | | +----- unused, fixed to 0
; | | | | | | | +----- unused, fixed to 0
; | | | | | | | +----- unused, fixed to 0

cTOC00ini   equ    00000001B           ;16-bit Timer Output Control Register
; | | | | | | | |
; | | | | | | | +----- TOE00, [0] disables, [1] enables outp
; | | | | | | | +----- TOC001,[0] disables [1] enables inv
; | | | | | | |
; | | | | | | |           LVS00  LVR00  Timer output F/F status
; | | | | | | | -----
; | | | | | | | +----- 0      0      no change
; | | | | | | |           0      1      Timer output F/F
; | | | | | | |           1      0      Timer output F/F set
; | | | | | | |           1      1      Setting prohibited
; | | | | | | |
; | | | | | | | +----- TOC004, [0]disables,[1]enables inv. op
; | | | | | | | +----- unused, fixed to 0
; | | | | | | | +----- unused, fixed to 0
; | | | | | | | +----- unused, fixed to 0

```

## Chapter 7 Program Listing

```

cPRM00ini    equ        00010010B                ;Prescaler Mode Register 00
; | | | | | | | |
; | | | | | | | |                PRM001  PRM000 count clock selection
; | | | | | | | |                -----
; | | | | | | | | ++----- 0      0      fx (10MHz)
; | | | | | | | |                0      1      fx/4 (2.5MHz)
; | | | | | | | |                1      0      fx/256 (39.06kHz)
; | | | | | | | |                1      1      TI000 valid edge
; | | | | | | | |
; | | | | | | | | +----- unused, fixed to 0
; | | | | | | | | +----- unused, fixed to 0
; | | | | | | | |
; | | | | | | | |                ES001  ES000  TI000 valid edge select
; | | | | | | | |                -----
; | | | | | | | | ++----- 0      0      falling edge
; | | | | | | | |                0      1      rising edge
; | | | | | | | |                1      0      setting prohibites
; | | | | | | | |                1      1      both edges
; | | | | | | | |
; | | | | | | | |                ES101  ES100  TI010 valid edge select
; | | | | | | | |                -----
; | | | | | | | | ++----- 0      0      no change
; | | | | | | | |                0      1      Timer output F/F
; | | | | | | | |                1      0      Timer output F/F set
; | | | | | | | |                1      1      Setting prohibited

;-----

InitTM00:    mov        TMC00,#cTMC00ini        ; count clock selection
             mov        CRC00,#cCRC00ini        ; select PWM mode, output enable
             mov        TOC00,#cTOC00ini
             mov        PRM00,#cPRM00ini

             ret

             END

```

## 7.3 Start-up

```

/*****
/*
/* File:      Startup.asm
/* Project:   Motor Control ( chopper ) - start-up
/*
/* Description: Contains power-on reset initialisation and the idle loop.*/
/*
/* History:   031103  creation
/*
*****/
; Enviroment      Device:      uPD78F0103
;                  Assembler:  A78000      V3.21A/386
;                  Compiler:   ICC78000    V3.21A/386
;                  Linker:     XLINK       V4.51J/386
;=====
;Changes
;=====

NAME      Startup      ; module name
; P780103      ; target device

#include   "df0103.h"   ; header file

;=====
; EXPORT LIST
;=====
; Constants:
; Variables:
; Flags:
; Labels:

PUBLIC   Startup

;=====
;IMPORT LIST
;=====
; Constants:
; Variables:
; Flags:
; Labels:

```

## Chapter 7 Program Listing

```
=====
; Segment name:  UDATA0
; Segment type:  relocatable data segment
; Address area:  short address RAM area
=====
RSEG      UDATA0  (1)                ; (1) segment starts at even address
SADDR

EVEN
startup_delay_in      DS      2
startup_delay_out     DS      1
startup_reference_v   DS      1
startup_measured_v    DS      1

=====
; Block type:      Standard Subprogram
; Purpose:         Startup
; Input:          none
; Output:         none
=====
RSEG      CODE
-----

Startup:  movw   AX,ADCR                ; AD , speed reference value
          mov    startup_reference_v,A ; store speed reference value
          mov    A,#0C0H                ; PWM start value (lower limit)
          setl   TCE50                  ; count operation start
          setl   TOE50                  ; output enable
cont_startup:mov  CR50,A                ; set with decremented PWM value
          cmp    A,startup_reference_v ; check startup stop value
          bz     retur                  ; if PWM=reference leave startup
          dec    A
          call   startup_delay          ; startup delay time constant
          mov    B,A
          movw   AX,ADCR                ; AD , speed reference value
          mov    startup_reference_v,A ; store speed reference value
          mov    A,B
          br     cont_startup

retur:    ret
```

```
;/=====
; Block type:          Standard Subroutine
; Purpose:            Startup Regulation Timing
; Input:              none
; Output:             none
;/=====

startup_delay: push   AX
                mov    startup_delay_out,#060H
loop_out:      nop
                movw   startup_delay_in,#6000H
loop_in:       dec    startup_delay_in
                nop
                BNZ    loop_in
                dec    startup_delay_out
                BNZ    loop_out
                pop    AX
                ret

                END
```

## 7.4 PWM Generation

```

/*****
/*
/* File:          PWM_Control.s26
/* Project:       Motor Control ( chopper ) - startup
/*
/*
/* Description:  Single phase Motor with chopper control
/*               This software example provides a basic PWM control algorithm
/*               for Universal Motor control. Timer_50 is used to generate a
/*               10kHz PWM signal. This PWM signal is switching via an IGBT
/*               the rectified line voltage and therewith the speed of the
/*               Universal Motor. Via a switch, it can be selected whether
/*               the system runs in closed or open loop mode.
/*               The speed can be controlled by a potmeter, connected to the
/*               microcontrollers 5V Vdd. The adjusted DC level is
/*               AD converted and therewith giving the reference speed value
/*               A magnetic ring with eight pol pairs is mounted on the motor
/*               rotor axis to measure the speed.
/*
/*
/*****
/*
/* History:       031103  creation
/*
/*
/*****
; Enviroment      Device:      uPD78F0103
;                  Assembler:  A78000      V3.21A/386
;                  Compiler:   ICC78000    V3.21A/386
;                  Linker:     XLINK       V4.51J/386
;=====
;Changes
;=====

NAME          PWM_Control      ; module name
; P780103     ; target device

;=====
; EXPORT LIST
;=====
; Constants:
; Variables:
; Flags:
; Labels:

PUBLIC      loop
PUBLIC      speed_reference_value
PUBLIC      speed_measured_value
PUBLIC      PWM_reference
PUBLIC      Monitor_ADCR
PUBLIC      Monitor_CR010
PUBLIC      Monitor_A
PUBLIC      Monitor_CR50
PUBLIC      speed_delta

```

## Chapter 7 Program Listing

```

;=====
; IMPORT LIST
;=====
; Constants:
; Variables:
; Flags:
; Labels:
EXTERN      Table      ; from Table

;=====
; INCLUDE FILE LIST
;=====

#include      "df0103.h"      ; header file
LSTOUT-
$df0103.h
LSTOUT+
COL 132

;=====
; Segment name:  UDATA0
; Segment type:  relocatable data segment
; Address area:  short address RAM area
;=====
RSEG      UDATA0  (1)      ; (1) segment starts even address
SADDR

;=====
; Global:

                                EVEN
Monitor_CR010      DS      2
Monitor_ADCR       DS      2
Monitor_CR50       DS      1
Monitor_A           DS      1
speed_reference_value DS      1
speed_measured_value DS      1
PWM_reference       DS      1
speed_delta        DS      1
                                EVEN
regulation_delay_const_in DS      2
startup_delay_in    DS      2
regulation_delay_const_out DS      1
startup_delay_out   DS      1

;=====
RSEG      Data      ; segment starts at even address
SADDR

;=====
; empty

```

## Chapter 7 Program Listing

```

;=====
; Segment name:  BITVARS
; Segment type:  relocatable bit segment
; Address area:  short address RAM area
;=====
RSEG          BITVARS          ; segment starts at even address
SADDR
;=====
; empty

;=====
; Segment name:  INTVEC
; Segment type:  interrupt vector
; Address area:  ROM, 0x0000...0x003f
;=====
COMMON      INTVEC          ; should be located at 0000H in *.xcl file
;=====
; empty

;=====
; Segment name:  CODE
; Segment type:  relocatable code segment
; Address area:  internal ROM area
;=====
RSEG          CODE
;=====

;=====
; Block type:    Standard Subroutine
; Purpose:       startup and closed loop regulation
; Input:         none
; Output:        none
;=====

loop:
reference_speed:  movw    AX,ADC          ; capture speed ref. value
                 movw    Monitor_ADCR,AX  ; ( watch function )
                 cmp     A,#015H
                 BC      set_to_USL      ; set to Upper Speed Limit
                 cmp     A,#080H
                 BNC     set_to_LSL      ; set to Lower Speed Limit
                 mov     speed_reference_value,A ; store speed ref. value
                 BR      measured_speed
set_to_USL:      mov     speed_reference_value,#015H
                 BR      measured_speed
set_to_LSL:      mov     speed_reference_value,#080H

measured_speed:  movw    AX,CR010        ; mov measured speed to AX
                 movw    Monitor_CR010,AX ; help word ( watch function )

                 xch     A,X
                 mov     speed_measured_value,A ; speed measured value

delta_speed:     sub     A,speed_reference_value
                 mov     speed_delta,A    ; speed difference
                 bz      speed_stable     ; delta=0, speed ok go to stable

```

## Chapter 7 Program Listing

```

                                bnc     speed_to_slow      ; C=1 means speed to high
speed_to_high:                 cmp     PWM_reference,#0C0H; check lower speed limit
                                bc      increment         ; if no, increment PWM value
                                mov     A,#0C0H           ; if yes, set lower speed limit
                                mov     PWM_reference,A
                                br      set_new_PWM

increment:                     inc     PWM_reference      ; inc PWM ref => decrease speed
                                inc     PWM_reference
                                inc     PWM_reference
                                cmp     PWM_reference,#0C0H ; check lower speed limit
                                bc      set_new_PWM        ;
                                mov     A,#0C0H           ; reset low. limit if overflow
                                mov     PWM_reference,A
                                br      set_new_PWM

speed_to_slow:                 cmp     PWM_reference,#020H; check if PWM below upper limit
                                bnc     decrement         ; if yes decrement PWM value
                                mov     A,#020H           ; if yes, set PWM to upper limit
                                mov     PWM_reference,A
                                br      set_new_PWM

decrement:                     dec     PWM_reference      ; dec PWM ref => increase speed
                                dec     PWM_reference
                                dec     PWM_reference
                                cmp     PWM_reference,#020H ; check upper speed limit
                                bnc     set_new_PWM        ;
                                mov     A,#020H           ; reset up. limit if underflow
                                mov     PWM_reference,A

set_new_PWM:                   mov     A,PWM_reference
                                BT      P12.0,c_loop      ; sel closed/open loop operation
                                mov     A,speed_reference_value
                                mov     B,A               ; speed ref to B for Table calc.
                                movw    HL,#Table
                                mov     A,[HL+B]          ; mov corresponding PWM to A

                                mov     P13,#1            ; generation of reference pulse
                                nop
                                mov     P13,#0

c_loop:                        mov     CR50,A            ; set new PWM_value
                                mov     Monitor_CR50,A    ; help register ( watch func. )
                                mov     PWM_reference,A    ; store new PWM_value
speed_stable:                   call    regulation_delay ; regulation time constant

                                ret
```

## Chapter 7 Program Listing

---

```
;=====
; Block type:      Standard Subroutine
; Purpose:         Adjustment Regulation Timing
; Input:           none
; Output:          none
;=====
regulation_delay:  mov regulation_delay_const_out,#01H; 02 0080 = 330usec
loop_reg_out:     nop
                  movwregulation_delay_const_in,#0040H
loop_reg_in:      dec regulation_delay_const_in
                  nop
                  BNZ loop_reg_in
                  dec regulation_delay_const_out
                  BNZ loop_reg_out
                  ret
                  END

; Note: setting 15-01-04 01_0040_upper 040_6000_lower
```

## 7.5 Display

```

/*****/
/*
/* File:          display.asm
/* Project:       Motor Control ( chopper ) - startup
/*
/* Description:  Contains power-on reset initialisation and the idle loop.*/
/*
/* History:      031103 creation
/*
/*****/
; Enviroment   Device:      uPD78F0103
;              Assembler:  A78000      V3.21A/386
;              Compiler:   ICC78000    V3.21A/386
;              Linker:     XLINK        V4.51J/386
;=====
;Changes
;=====

NAME          Display          ; module name
; P780103     ; target device

#include       "df0103.h"      ; header file

;=====
; EXPORT LIST
;=====
; Constants:
; Variables:
; Flags:
; Labels:

PUBLIC       Display

;=====
; IMPORT LIST
;=====
; Constants:
EXTERN      speed_measured_value; Variables:
EXTERN      speed_reference_value
; Flags:
; Labels:

EXTERN      PWM_generation    ; from .....
EXTERN      InitMicro        ; from Micro
EXTERN      loop              ; from PWM_generation

;=====
; Segment name:  CODE
; Segment type:  relocatable code segment
; Address area:  internal ROM area
;=====
RSEG        CODE
;=====

```

## Chapter 7 Program Listing

```
=====
; Block type:          Subprogram
; Purpose:             Display speed error
; Input:              none
; Output:             none
=====
Display:                ; clear ports

    mov    P3,#00H
    clr1  P0.1
    clr1  P0.2
    clr1  P0.3
    clr1  P1.0
    clr1  P1.6
    clr1  P1.5
    clr1  P1.4

    BF    P12.0,skip    ; check open / closed loop

    mov    A,speed_measured_value ; caculate speed difference
    sub    A,speed_reference_value
    bz     GreenLED
    bnc   to_slow

to_fast:    cmp    A,#0F9H                ; select accuracy LED
            bnc   LEDP03
            cmp    A,#0F3H
            bnc   LEDP02
            cmp    A,#0EDH
            bnc   LEDP01
            cmp    A,#0E7H
            bnc   LEDP10
            br    Y_fast

to_slow:    cmp    A,#07H                ; select accuracy LED
            bc    LEDP30
            cmp    A,#0DH
            bc    LEDP31
            cmp    A,#013H
            bc    LEDP32
            cmp    A,#019H
            bc    LEDP33
            br    Y_slow

Y_slow:    set1  P1.5                    ; accuracy > 8%
            ret

Y_fast:    set1  P1.4                    ; accuracy > 8%
            ret

GreenLED:  set1  P1.6
            ret

LEDP30     set1  P3.0
            ret

LEDP31     set1  P3.1
            ret

LEDP32     set1  P3.2
            ret
```

## Chapter 7 Program Listing

---

```
LEDP33      set1  P3.3
            ret
LEDP03      set1  P0.3
            ret
LEDP02      set1  P0.2
            ret
LEDP01      set1  P0.1
            ret
LEDP10      set1  P1.0
skip        ret

            END
```

## 7.6 Conversion Table

```

; Motor Control
; Table
; Conversion Table to adapt Motor speed in open / closed loop operation

```

```

PUBLIC          Table

RSEG          CONST
saddr

```

```
Table:
```

```

;AD 00h
    DB  018H    ; AD_00
    DB  018H    ; AD_01
    DB  018H    ; AD_05
    DB  018H    ; AD_05
    DB  018H    ; AD_05
    DB  018H    ; AD_05
    DB  018H    ; AD_06
    DB  018H    ; AD_07
    DB  018H    ; AD_08
    DB  018H    ; AD_09
    DB  018H    ; AD_0A
    DB  018H    ; AD_0B
    DB  018H    ; AD_0C
    DB  018H    ; AD_0B
    DB  018H    ; AD_0E
    DB  018H    ; AD_0F

;AD 10h
    DB  018H    ; AD_10
    DB  019H    ; AD_11
    DB  019H    ; AD_15
    DB  020H    ; AD_15
    DB  020H    ; AD_15
    DB  022H    ; AD_15
    DB  028H    ; AD_16
    DB  02BH    ; AD_17
    DB  030H    ; AD_18
    DB  035H    ; AD_19
    DB  03CH    ; AD_1A
    DB  045H    ; AD_1B
    DB  050H    ; AD_1C
    DB  058H    ; AD_1B
    DB  05EH    ; AD_1E
    DB  064H    ; AD_1F

;AD 20h
    DB  064H    ; AD_20
    DB  066H    ; AD_21
    DB  068H    ; AD_22
    DB  069H    ; AD_23
    DB  070H    ; AD_24
    DB  072H    ; AD_25
    DB  073H    ; AD_26
    DB  074H    ; AD_27
    DB  075H    ; AD_28

```

## Chapter 7 Program Listing

---

```
DB 076H ; AD_29
DB 077H ; AD_2A
DB 078H ; AD_2B
DB 080H ; AD_2C
DB 082H ; AD_2B
DB 084H ; AD_2E
DB 085H ; AD_2F

;AD 30h
DB 085H ; AD_30
DB 086H ; AD_31
DB 087H ; AD_33
DB 088H ; AD_33
DB 089H ; AD_33
DB 089H ; AD_33
DB 08AH ; AD_36
DB 08AH ; AD_37
DB 08BH ; AD_38
DB 08CH ; AD_39
DB 08DH ; AD_3A
DB 08EH ; AD_3B
DB 090H ; AD_3C
DB 092H ; AD_3B
DB 094H ; AD_3E
DB 096H ; AD_3F

;AD 40h
DB 096H ; AD_40
DB 097H ; AD_41
DB 098H ; AD_43
DB 099H ; AD_43
DB 09AH ; AD_44
DB 09AH ; AD_45
DB 09AH ; AD_46
DB 09BH ; AD_47
DB 09BH ; AD_48
DB 09BH ; AD_49
DB 09CH ; AD_4A
DB 09CH ; AD_4B
DB 09CH ; AD_4C
DB 09DH ; AD_4B
DB 09EH ; AD_4E
DB 0A0H ; AD_4F

;AD 50h
DB 0A0H ; AD_50
DB 0A1H ; AD_51
DB 0A1H ; AD_55
DB 0A2H ; AD_55
DB 0A2H ; AD_55
DB 0A2H ; AD_55
DB 0A3H ; AD_56
DB 0A3H ; AD_57
DB 0A4H ; AD_58
DB 0A4H ; AD_59
DB 0A5H ; AD_5A
DB 0A6H ; AD_5B
DB 0A7H ; AD_5C
```

## Chapter 7 Program Listing

---

```
DB 0A7H ; AD_5B
DB 0A8H ; AD_5E
DB 0A8H ; AD_5F

;AD 60h
DB 0A8H ; AD_60
DB 0A8H ; AD_61
DB 0A8H ; AD_62
DB 0A9H ; AD_63
DB 0A9H ; AD_64
DB 0A9H ; AD_65
DB 0AAH ; AD_66
DB 0AAH ; AD_67
DB 0AAH ; AD_68
DB 0ABH ; AD_69
DB 0ACH ; AD_6A
DB 0ACH ; AD_6B
DB 0ADH ; AD_6C
DB 0ADH ; AD_6B
DB 0ADH ; AD_6E
DB 0AEH ; AD_6F

;AD 70h
DB 0AEH ; AD_70
DB 0AEH ; AD_71
DB 0AFH ; AD_72
DB 0AFH ; AD_73
DB 0B0H ; AD_74
DB 0B1H ; AD_75
DB 0B1H ; AD_76
DB 0B2H ; AD_77
DB 0B2H ; AD_78
DB 0B2H ; AD_79
DB 0B3H ; AD_7A
DB 0B3H ; AD_7B
DB 0B3H ; AD_7C
DB 0B4H ; AD_7B
DB 0B4H ; AD_7E
DB 0B4H ; AD_7F

;AD 80h
DB 0B4H ; AD_80
DB 0B5H ; AD_81
DB 0B5H ; AD_82
DB 0B5H ; AD_83
DB 0B6H ; AD_84
DB 0B6H ; AD_85
DB 0B6H ; AD_86
DB 0B6H ; AD_87
DB 0B6H ; AD_88
DB 0B7H ; AD_89
DB 0B7H ; AD_8A
DB 0B7H ; AD_8B
DB 0B7H ; AD_8C
DB 0B7H ; AD_8B
DB 0B7H ; AD_8E
DB 0B8H ; AD_8F
```

```
;AD 90h
DB 0B8H ; AD_90
DB 0B8H ; AD_91
DB 0B8H ; AD_92
DB 0B8H ; AD_93
DB 0B8H ; AD_94
DB 0B9H ; AD_95
DB 0B9H ; AD_96
DB 0B9H ; AD_97
DB 0BAH ; AD_98
DB 0BAH ; AD_99
DB 0BAH ; AD_9A
DB 0BAH ; AD_9B
DB 0BAH ; AD_9C
DB 0BBH ; AD_9B
DB 0BBH ; AD_9E
DB 0BBH ; AD_9F
```

```
;AD A0h
DB 0BBH ; AD_A0
DB 0BBH ; AD_A1
DB 0BBH ; AD_A2
DB 0BCH ; AD_A3
DB 0BCH ; AD_A5
DB 0BCH ; AD_A6
DB 0BCH ; AD_A7
DB 0BCH ; AD_A8
DB 0BCH ; AD_A9
DB 0BDH ; AD_AA
DB 0BDH ; AD_AB
DB 0BDH ; AD_AC
DB 0BDH ; AD_AB
DB 0BDH ; AD_AE
DB 0BDH ; AD_AF
```

```
;AD B0h
DB 0BDH ; AD_B0
DB 0BEH ; AD_B1
DB 0BEH ; AD_B2
DB 0BEH ; AD_B3
DB 0BEH ; AD_B4
DB 0BFH ; AD_B5
DB 0BFH ; AD_B6
DB 0BFH ; AD_B7
DB 0C0H ; AD_B8
DB 0C0H ; AD_B9
DB 0C0H ; AD_BA
DB 0C0H ; AD_BB
DB 0C0H ; AD_BC
DB 0C0H ; AD_BB
DB 0C0H ; AD_BE
DB 0C0H ; AD_BF
```

```
;AD C0h
DB 0C0H ; AD_C0
DB 0C0H ; AD_C1
DB 0C0H ; AD_C2
```

## Chapter 7 Program Listing

---

```
DB 0C0H ; AD_C3
DB 0C0H ; AD_C4
DB 0C0H ; AD_C5
DB 0C0H ; AD_C6
DB 0C0H ; AD_C7
DB 0C2H ; AD_C8
DB 0C2H ; AD_C9
DB 0C4H ; AD_CA
DB 0C6H ; AD_CB
DB 0C8H ; AD_CC
DB 0CAH ; AD_CB
DB 0CBH ; AD_CE
DB 0D0H ; AD_CF
```

;AD D0h

```
DB 0D0H ; AD_D0
DB 0D0H ; AD_D1
DB 0D0H ; AD_D2
DB 0D0H ; AD_D3
DB 0D0H ; AD_D4
DB 0D0H ; AD_D5
DB 0D0H ; AD_D6
DB 0D0H ; AD_D7
DB 0D0H ; AD_D8
DB 0D0H ; AD_D9
DB 0D0H ; AD_DA
DB 0D0H ; AD_DB
DB 0D0H ; AD_DC
DB 0D0H ; AD_DB
DB 0D0H ; AD_DE
DB 0D0H ; AD_DF
```

;AD E0h

```
DB 0D0H ; AD_E0
DB 0D0H ; AD_E1
DB 0D0H ; AD_E2
DB 0D0H ; AD_E3
DB 0D0H ; AD_E4
DB 0D0H ; AD_E5
DB 0D0H ; AD_E6
DB 0D0H ; AD_E7
DB 0D0H ; AD_E8
DB 0D0H ; AD_E9
DB 0D0H ; AD_EA
DB 0D0H ; AD_EB
DB 0D0H ; AD_EC
DB 0D0H ; AD_EB
DB 0D0H ; AD_EE
DB 0D0H ; AD_EF
```

;AD F0h

```
DB 0D0H ; AD_F0
DB 0D0H ; AD_F1
DB 0D0H ; AD_F2
DB 0D0H ; AD_F3
DB 0D0H ; AD_F4
DB 0D0H ; AD_F5
DB 0D0H ; AD_F6
```

```
DB 0D0H ; AD_F7
DB 0D0H ; AD_F8
DB 0D0H ; AD_F9
DB 0D0H ; AD_FA
DB 0D0H ; AD_FB
DB 0D0H ; AD_FC
DB 0D0H ; AD_FB
DB 0D0H ; AD_FE
DB 0D0H ; AD_FF
```

```
end
```

## Facsimile Message

From:

\_\_\_\_\_  
Name

\_\_\_\_\_  
Company

\_\_\_\_\_  
Tel.

\_\_\_\_\_  
FAX

\_\_\_\_\_  
Address

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

*Thank you for your kind support.*

**North America**

NEC Electronics America Inc.  
Corporate Communications Dept.  
Fax: 1-800-729-9288  
1-408-588-6130

**Hong Kong, Philippines, Oceania**

NEC Electronics Hong Kong Ltd.  
Fax: +852-2886-9022/9044

**Asian Nations except Philippines**

NEC Electronics Singapore Pte. Ltd.  
Fax: +65-6250-3583

**Europe**

NEC Electronics (Europe) GmbH  
Market Communication Dept.  
Fax: +49(0)-211-6503-1344

**Korea**

NEC Electronics Hong Kong Ltd.  
Seoul Branch  
Fax: 02-528-4411

**Japan**

NEC Semiconductor Technical Hotline  
Fax: +81- 44-435-9608

**Taiwan**

NEC Electronics Taiwan Ltd.  
Fax: 02-2719-5951

I would like to report the following error/make the following suggestion:

Document title: \_\_\_\_\_

Document number: \_\_\_\_\_ Page number: \_\_\_\_\_

If possible, please fax the referenced page or drawing.

Document Rating	Excellent	Good	Acceptable	Poor
Clarity	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Technical Accuracy	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Organization	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

[MEMO]