To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas 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 Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "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.

   "High Quality": 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; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

RENESAS

Application Note

78K0 and 78K0S

Low Cost Triac Control using NEC 8-bit Microcontrollers

Document No. U16498EE1V1AN00
Date Published May 2004

© NEC Corporation 2004
Printed in Germany

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 10.05. 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

Chapter 1 Overview 11
- 1.1 Introduction 11
- 1.2 Overview of the 78K0 and 78K0S Families 12

Chapter 2 Triac Control 15
- 2.1 Triac Control Basics 15
- 2.2 Triac Triggering Basics 18

Chapter 3 System Design 21
- 3.1 System Overview 21
- 3.2 Power Supply 22
- 3.3 Zero Crossing Detector 23

Chapter 4 Hardware Configuration. 25
- 4.1 Schematic Diagrams. 25
- 4.2 Device Configurations 26
- 4.3 Peripherals and I/O Usage 27
- 4.4 16-Bit Timer Function 29
- 4.5 8-Bit Timer Function 29
- 4.6 Watchdog Timer Function 29
- 4.7 A/D converter 30
- 4.8 Interrupt Functions 30

Chapter 5 Software Process Descriptions. 31
- 5.1 Initialisation 31
- 5.2 CSTARTUP 31
- 5.3 A_to_D 31
- 5.4 Main 32
- 5.5 Zero_Cross (INTP0 - Interrupt Service Routine) 33
- 5.6 Phase_Firing (8-bit Timer Compare Match ISR). 34

Chapter 6 Conclusions 35

Chapter 7 Flow-Charts. 37
- 7.1 Main Flow Diagram 37
- 7.2 Zero Cross - ISR 38
- 7.3 Phase Firing - ISR 39
- 7.4 A_to_D 40

Appendix A 78K0 Program Listings 41

Appendix B 78K0S Program Listings 47

List of Figures

Figure 2-1: Basic Current and Voltage Relationship 15
Figure 2-2: Phase Angle and Load Current Relationships 16
Figure 2-3: Basic Timing Relationships 17
Figure 2-4: Triac Quadrant Definitions 19
Figure 3-1: System Block Diagram 21
Figure 3-2: Basic R/C "Dropper" Power Supply 22
Figure 3-3: Example Zero Crossing Detector 23
Figure 4-1: µPD78F0103 Schematic Diagram 25
Figure 4-2: µPD78F9116 Schematic Diagram 26

List of Tables

Table 1-1: 78K0S Series, Differences amongst the basic Subseries 12
Table 1-2: 78K0 Series, Differences amongst the basic Subseries..... 13
Table 4-1: Device Configurations 26
Table 4-2: µPD78F9116 I/O Assignment..... 27
Table 4-3: µPD78F0103 I/O Assignment..... 28

Chapter 1 Overview

1.1 Introduction

This Application Note serves as an example of a Low Cost Triac control of a Universal AC motor using NEC K0 and K0S 8 Bit Microcontrollers.

The aim of this application note is to provide the user with an example of driving of a Triac directly from the mains, using a low cost Microcontroller.

In particular, we will describe a possible solution to drive a load with the Phase Angle Partialisation method.

1.2 Overview of the 78K0 and 78K0S Families

The 78K0S and 78K0, 8 bit families are highly integrated single-chip Microcontrollers. They feature CPU, ROM, RAM and peripheral functions such as A/D-converter, UART, I²C and CSI Serial interfaces as well as many dedicated ASSP peripheral functions. All offer a range of on chip memory options in both Flash and Masked ROM.

The heart of the 78K0 and 78K0S families are a powerful 8/16-bit CPU providing an excellent power / performance ratio.

The 78K0 core provides four register banks each with eight 8-bit general registers.
Registers 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.

The78K0S core provides a single Register bank, consisting of eight 8 bit general registers. Registers can be concatenated to form 16bit registers, supporting 16-bit operations. Bit manipulation operations are supported on all registers and the entire RAM address space. A reduced set of 47 optimised instructions is available, which are upwardly compatible with the 78K0 families.

Both families offer options for Sub-Clock operation to reduce system power consumption.

Table 1-1: 78K0S Series, Differences amongst the basic Subseries

<table>
<tr><th colspan="2">Function / Subseries Name</th><th>ROM Capacity</th><th colspan="4">Timer</th><th>8-bit A/D</th><th>10-bit A/D</th><th>Serial Interface</th><th>I/O</th><th>V_{DD} MIN value</th><th>Remark</th></tr>
<tr><th></th><th></th><th></th><th>8-bit</th><th>16-bit</th><th>WT</th><th>WDT</th><th></th><th></th><th></th><th></th><th></th><th></th></tr>
<tr><td rowspan="6">Small-scale general purpose</td><td>µPD789046</td><td>16K</td><td rowspan="3">1 ch</td><td rowspan="4">1 ch</td><td>1 ch</td><td rowspan="5">1 ch</td><td rowspan="6">-</td><td rowspan="6">-</td><td rowspan="4">1 ch (UART: 1 ch)</td><td rowspan="2">34</td><td rowspan="5">1.8 V</td><td rowspan="4">-</td></tr>
<tr><td>µPD789026</td><td>4K to 16K</td><td rowspan="4">-</td></tr>
<tr><td>µPD789074</td><td>2K to 8K</td></tr>
<tr><td>µPD789088</td><td>16K to 32K</td><td>3 ch</td><td rowspan="2">24</td></tr>
<tr><td>µPD789062</td><td rowspan="2">4K</td><td rowspan="2">2 ch</td><td rowspan="2">-</td><td rowspan="2">-</td><td>14</td><td>RC oscillation</td></tr>
<tr><td>µPD789052</td><td>-</td></tr>
<tr><td rowspan="10">Small-scale general purpose + A/D</td><td>µPD789177Y</td><td rowspan="4">16K to 24K</td><td rowspan="4">3 ch</td><td rowspan="6">1 ch</td><td rowspan="4">1 ch</td><td rowspan="6">1 ch</td><td>-</td><td>8 ch</td><td rowspan="2">2 ch (UART: 1 ch) (SMB: 1 ch)</td><td rowspan="4">31</td><td rowspan="10">1.8 V</td><td rowspan="6">-</td></tr>
<tr><td>µPD789167Y</td><td>8 ch</td><td>-</td></tr>
<tr><td>µPD789177</td><td>-</td><td>8 ch</td><td rowspan="6">1 ch (UART: 1 ch)</td></tr>
<tr><td>µPD789167</td><td>8 ch</td><td>-</td></tr>
<tr><td>µPD789134A</td><td rowspan="4">2K to 8K</td><td rowspan="4">1 ch</td><td>-</td><td>4 ch</td><td rowspan="4">20</td><td rowspan="2">RC oscillation</td></tr>
<tr><td>µPD789124A</td><td rowspan="3">-</td><td>4 ch</td><td>-</td></tr>
<tr><td>µPD789114A</td><td>-</td><td>4 ch</td><td rowspan="2">-</td></tr>
<tr><td>µPD789104A</td><td>4 ch</td><td>-</td></tr>
<tr><td>Inverter controller</td><td>µPD789842</td><td>8K to 18K</td><td>3 ch</td><td>-</td><td>1 ch</td><td>1 ch</td><td>8 ch</td><td>-</td><td>1 ch (UART: 1 ch)</td><td>30</td><td>4.0 V</td><td>-</td></tr>
</table>

Remark: This table represents a small subsection of the 78K0S 8-bit Microcontroller family.

Table 1-2: 78K0 Series, Differences amongst the basic Subseries

Subseries Name \ Function		ROM Capacity	Timer				8-bit A/D	10-bit A/D	8-bit D/A	Serial Interface	I/O	V_{DD} MIN value	External expansion
			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^2C: 1 ch)			
	μPD78078Y												
	μPD78070A	-								3 ch (UART: 1 ch)	61	2.7 V	
	μPD78070AY									3 ch (UART: 1 ch, I^2C: 1 ch)			
	μPD780058	24K to 60K	2 ch							3 ch (time-division UART: 1 ch)	68	1.8 V	
	μPD780058Y									3 ch (time-division UART: 1 ch, I^2C: 1 ch)			
	μPD780065	40K to 48K							-	4 ch (UART: 1 ch)	60	2.7 V	
	μPD780078	48K to 60K		2 ch			-	8 ch		4 ch (UART: 1 ch, I^2C: 1 ch)	52	1.8 V	
	μPD780078Y												
	μPD780034AS	8K to 32K		1 ch				4 ch		3 ch (UART: 1 ch, time-division 3-wire: 1 ch)	39		-
	μPD780034A							8 ch			51		×
	μPD780034AY						-	8 ch		3 ch (UART: 1 ch, I^2C: 1 ch)			
	μPD780024AY						8 ch	-					
On-chip power-on reset	78K0/KB1	8K to 24K	3 ch	1 ch	-	1 ch	-	4 ch	-	3 ch (UART: 2 ch)	22	2.7 V	-
	78K0/KC1	8K to 32K	4 ch		1 ch			8 ch			32		
	78K0/KD1										39		
	78K0/KE1	8K to 60K		2 ch						4 ch (UART: 2 ch)	22		
	78K0/KF1	24K to60K								5 ch (UART: 2 ch)	22		×

Remark: This table represents a small subsection of the 78K0 8-bit Microcontroller family.

[MEMO]

Chapter 2 Triac Control

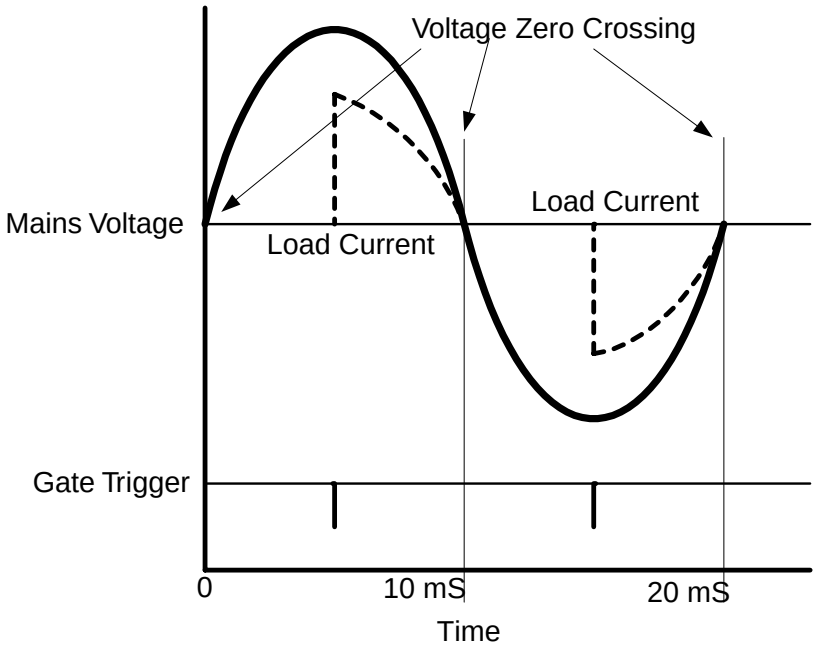
2.1 Triac Control Basics

A Triac is the optimum device for solid state control of Single Phase AC loads at low frequency, (i.e. mains.). Control can take the form of simple on-off switching as required for basic low cost equipment, such as thermostats or any simple loads where only full power must be applied.

It can also take the form of variable power control by the use of “phase control”, where the AC sine wave is chopped by delaying the Triac trigger in each half cycle of the mains cycle.

The basic Voltage / Current relationship for a resistive Load is shown in Figure 2-1.

Figure 2-1: Basic Current and Voltage Relationship

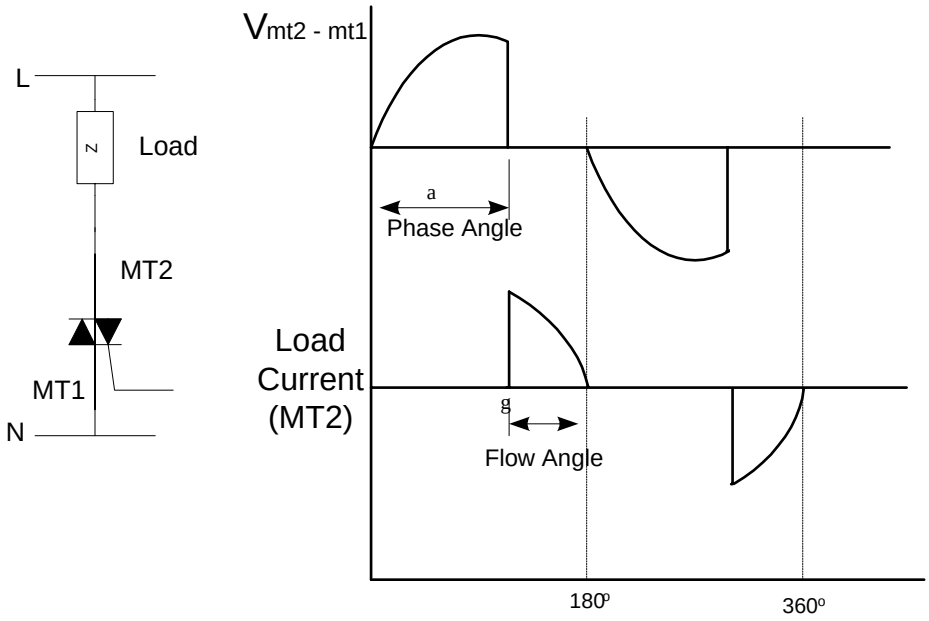


With a phase control system, the Triac is controlled during each period of the mains voltage. The power transferred to the load is proportional to the Current Flow. This makes this approach suitable to drive an inductive load (i.e. Universal or Single phase motors).

Figure 2-2 shows the relation between the Phase Angle "a" and the current flow angle "g", for an Inductive Load where the load Current lags the Voltage by 90°.

Figure 2-2: Phase Angle and Load Current Relationships

(Where MT1 and MT2 are Main Terminals 1 and 2 respectively)



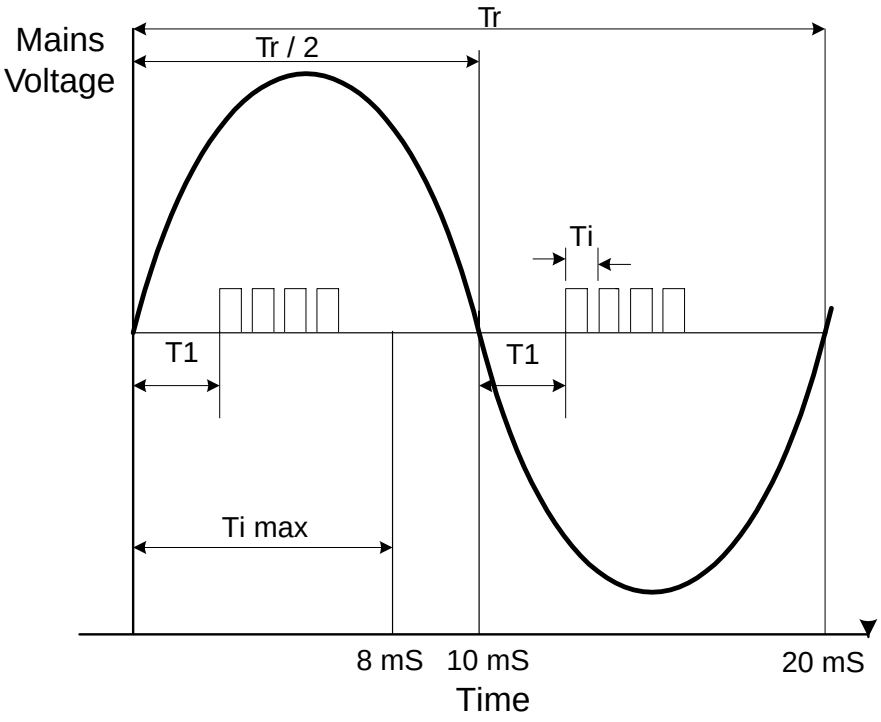
Increasing or decreasing the phase angle or more specifically the time delay during each mains cycle controls the motor speed. It is important therefore that the Phase Angle (time delay) is synchronised with the mains cycle and it needs to be kept constant during both half cycles. This means that, if the mains voltage frequency is 50 Hz the phase angle may be changed every 20 mS.
The Triac will turn off (“commutate”), when the load current reaches zero, thus the need to provide a phase angle trigger the in both halves of the Mains cycle.

The usual method of synchronising to the Mains cycle is to detect the Voltage Zero Crossing points. This is normally done for either the Positive or both Positive and Negative zero crossing points. Current zero crossing detection is normally used where a continuous supply is applied to the Load and the Triac is triggered at each current zero crossing point.

The Microcontroller can use the Voltage zero crossing points as a reference to calculate the appropriate Phase Angle (Time delay) requirements. Some allowance needs to be made for Inductive loads as current and voltage are not in phase. Ideally the trigger should be made at a time after the Load current has reached zero in order to reduce load current discontinuity and reduce Electromagnetic Interference (EMI).

Figure 2-3 shows a typical timing relationship to the 50 Hz mains cycle.

Figure 2-3: Basic Timing Relationships



Remarks: **1.** Tr is the Full Mains cycle time. (Based on 50 Hz)

2. Tr / 2 is the Half cycle time. (Based on 50 Hz)

3. T1 is the Phase angle delay

4. Ti max is the maximum Phase Angle delay (Ti max is used to avoid re-triggering the Triac in the next half cycle.)

5. Ti is the Triac Gate trigger pulse period

2.2 Triac Triggering Basics

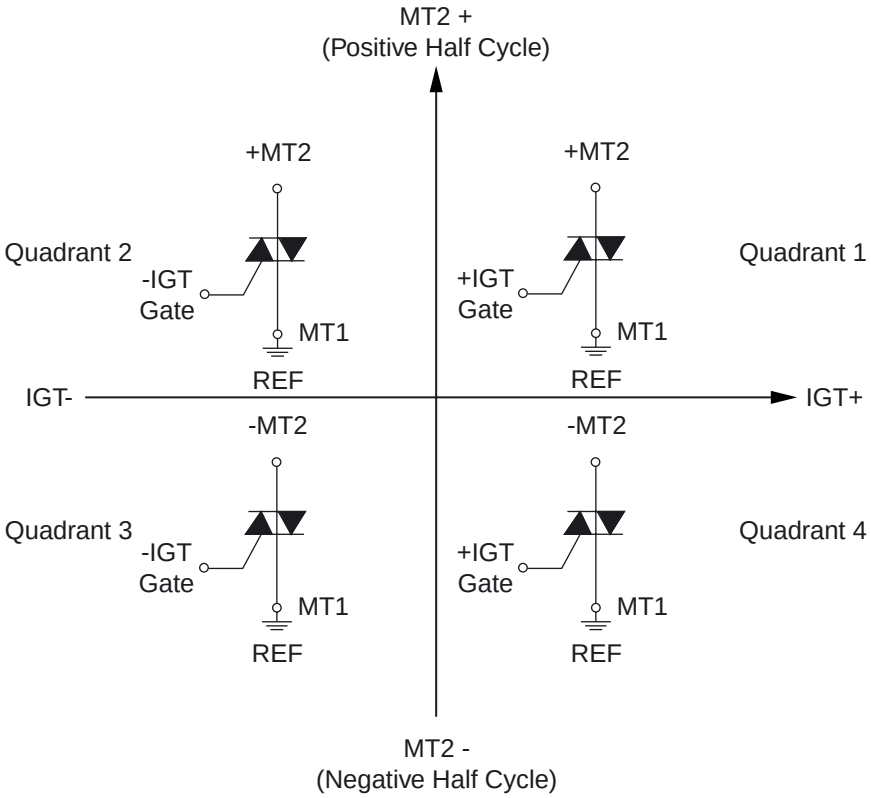
Triac's are manufactured in many different forms that include

- Three and Four Quadrant
- Standard and Snubberless
- Low Sensitivity

Three-quadrant Triac's are considered in this Application for their improved immunity to commutation failure or loss of control and false triggering. This is because a three-quadrant Triac does not trigger in the 4th quadrant (See Figure 2-4), so that the gate drive circuit cannot falsely trigger the Triac. This is not important as the vast majority of IC drive circuits, are designed to sink current. This certainly holds true for many Microcontrollers, which are capable of sinking more current than they can source.
In these cases Gate triggering is usually performed in Quadrants 2 and 3 (MT2+, G- and MT2-, G-), while triggering is not achieved in quadrant 1, as there is insufficient current available to trigger the Triac and the Gate Voltage is held in the inactive state.

Figure 2-4: Triac Quadrant Definitions

(Where MT1 and MT2 are Main Terminals 1 and 2 respectively)



Triac's that provide Low Gate Sensitivity are essential for applications that are driven directly from a Microcontroller, as they require low Gate trigger currents. Many Triac types are available on the market with gate trigger currents ranging from 5mA to 50mA. It is necessary to ensure that the Microcontroller supplies enough gate current for the Triac to exceed the "Latching" current through the Load. Once this has been reached the Gate current can be removed and the Triac will continue to conduct until the load current reaches zero (i.e. the Triac will commutate). Typically a number of Gate Trigger pulses are employed over a short period of time, rather than a continuous Gate Current, to ensure the latching current level is reached.

Historically a Triac that required low Gate trigger current also had a low Rate of Change of Voltage (dv/dt). In these cases a R/C "Snubber" network was usually required to ensure that the Triac and Load can be turned off (Especially for Inductive Loads). Also in some cases a series inductor with the Triac / Load was also needed, adding further complexity and component cost.

Triac's are generally identified as

- Standard (May require a Snubber circuit)
- Snubberless (Do not require a Snubber circuit)

Many modern Triac's however now provide a sufficiently high dv/dt specifications such that they do not require a "Snubber" or series inductor for inductive loads.

Two examples of Snubberless Three Quadrant Triac's with low gate sensitivity suitable to be directly driven from a Microcontroller are the BTA2xx Series, or MAC4DC series.

[MEMO]

Chapter 3 System Design

3.1 System Overview

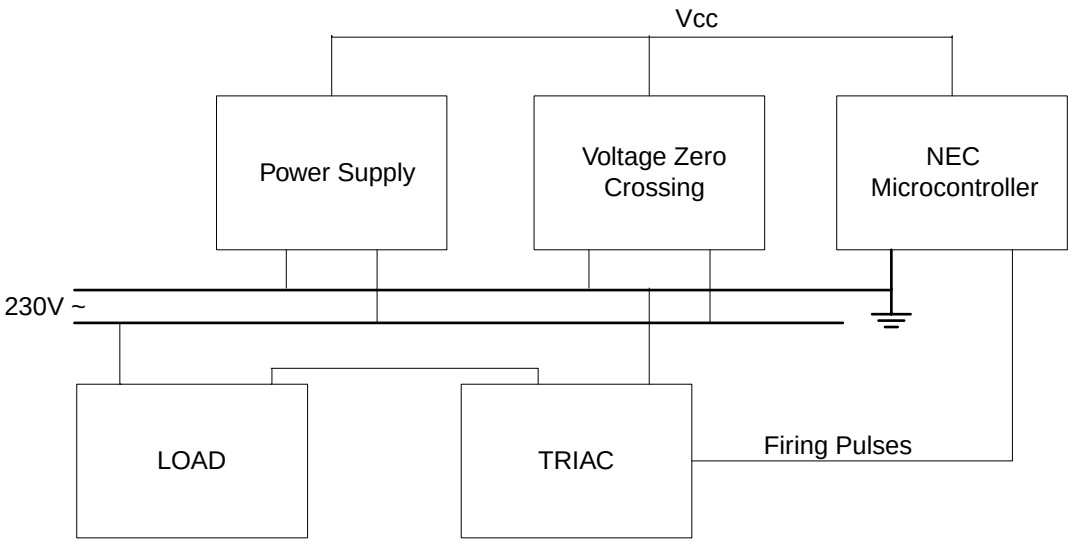
The basic goal of the design is to provide an example of a Triac control circuit for both 78K0 and 78K0S families, while keeping the design and software as portable as possible to other member of the 8 bit family.

The following devices were selected for this application example

- 78K0 µPD78F0103
- 78K0S µPD78F9116

The design is based on a Three-Quadrant Triac, using Phase Partialisation control with Negative Gate Trigger. The Gate is driven directly form the Microcontroller via a current limiting resister. The basic system diagram is shown in Figure 3-1.

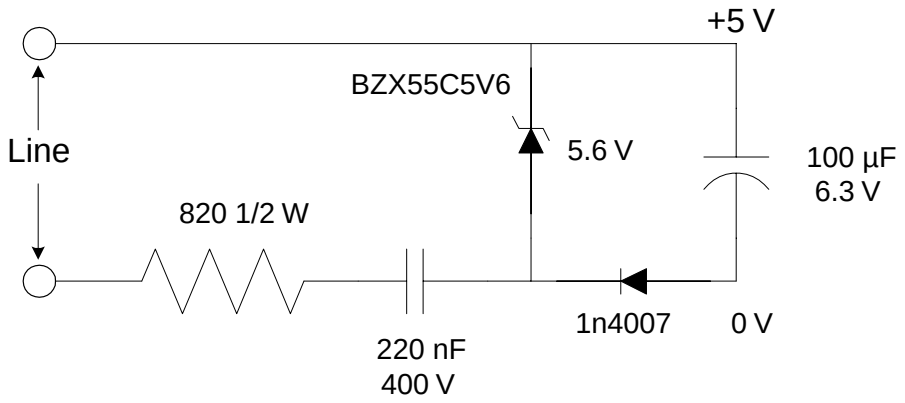
Figure 3-1: System Block Diagram



3.2 Power Supply

As the power requirements of the Microcontroller and Triac Gate currents are generally very low, power can be provided directly from the mains using a low cost power supply circuit. An example is shown in Figure 3-2. If the design requires a more protected or stable supply, then external high voltage integrated power supply devices are available.

Figure 3-2: Basic R/C "Dropper" Power Supply



The basic operation of this power supply design is based on a simple half wave rectifier where the load Capacitor is charged for only half of the mains cycle. It is therefore necessary to ensure that the circuit can supply enough current to the application without a significant voltage drop, during the mains half cycle where no charging is available. It is necessary to design the Power supply for twice the required current to allow for the phase where power is taken from the reservoir Capacitor.

The available current requirements can be calculated from the following equation:

$$I_{230V} = \frac{230\ V}{\sqrt{R1^2 + \left(\frac{1}{2\ \pi \times 50\ Hz \times C_{Dropper}}\right)^2}}$$

In the example shown in Figure 3-2, the power supply provides around 16 mA, which is more than enough for the Microcontroller. It depends largely on the average Triac Gate current as to if these component values are sufficient.
For example, a Triac with a Gate current of 35 mA that is triggered for 1 mS in each mains half period (i.e. every 10 mS), would produce an average current of 3.5 mA over the complete mains cycle. Thus making the maximum requirements for the power supply 8.5 mA including the Microcontroller.

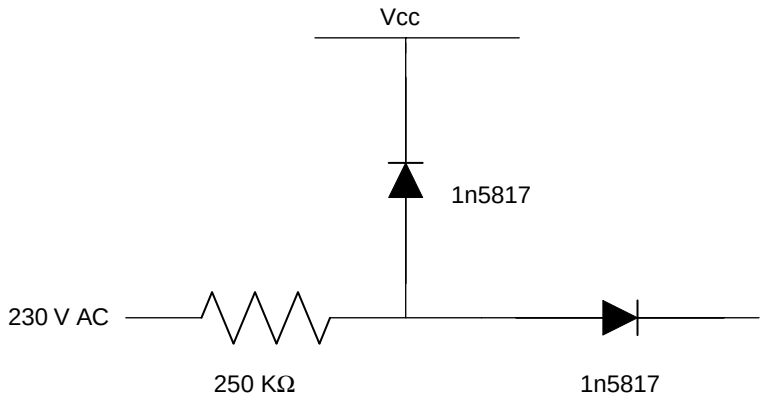
Typical Currents for the components used in this example are shown below.

- µPD78F0103A - 5 mA at 5 V and 4.19 MHz
- µPD78F9116B - 3 mA at 5 V and 4.19 MHz
- Typical Gate Current ≤ 35 mA

3.3 Zero Crossing Detector

In this example, we consider Voltage Zero Crossing Detection for synchronisation to the Mains**.**
There are many approaches to the detection but in this application example, it is we utilise an external discrete circuit such as the one shown below. An alternative would be to use an NEC Microcontroller with an on chip analogue comparator. This would allow a more accurate detector to be made, compensating for the unequal voltage switching thresholds of a standard CMOS I/O pin.
However here we consider the more common approach and utilise a standard input pin for the Zero Crossing input. This generates an internal Microcontroller Interrupt as the main timing reference. An example of a Zero Crossing circuit is shown in Figure 3-3.

Figure 3-3: Example Zero Crossing Detector



In this example design it was decided to provide a Crossing detection for both the Positive Zero crossing and Negative Zero crossing, in order to minimise the use of the on board timers in the Microcontroller and maintain a reasonable resolution for the Phase Angle delay. Alternatively only positive zero crossing detection can be made, where the timer is used to provide the time delay to the Triac triggering point (Phase Angle), for both positive and negative halves of the mains cycle.
The Zero crossing detector would normally be used to generate an external Interrupt to the Microcontroller, providing synchronisation to the mains supply.
Using both Positive and Negative crossing points has one limitation in that the resulting detection produces unequal timing reference points. Typically the first half cycle is approximately 8.5 mS and the second is approximately 11.5 mS for a 50 Hz supply. The reason for this is that a standard CMOS input pin is used for the zero crossing detection, the Low to High and High to Low threshold voltages are not equal and thus switch at different times.
This can be compensated for in the software to avoid triggering the Triac at the wrong point in the second half cycle.

The series resistor is used to limit the input current into the pin of the Microcontroller. All Microcontrollers can only accept a certain input current (Injected current), when the input voltage is above or below the supply or Ground references. The NEC devices can withstand an average 0.5 mA of injected current without any damage or long term reliability issues with the device.
In this circuit the average current is below 0.5 mA with the 250 KΩ resistor.

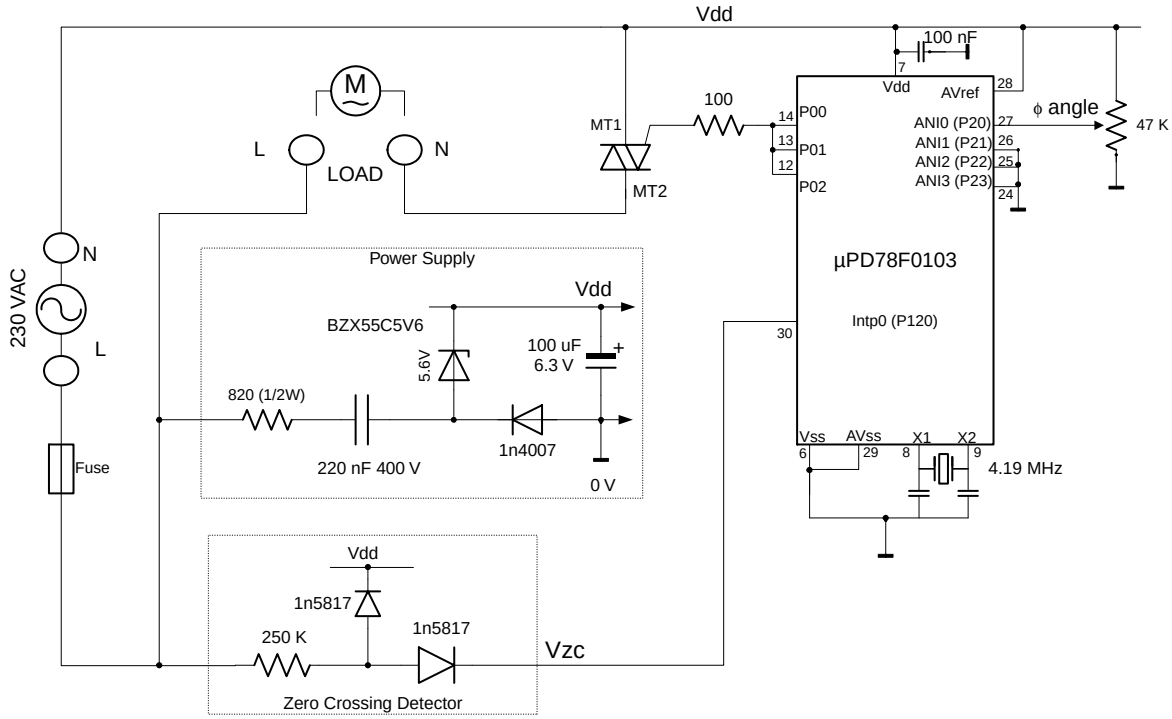
[MEMO]

Chapter 4 Hardware Configuration

4.1 Schematic Diagrams

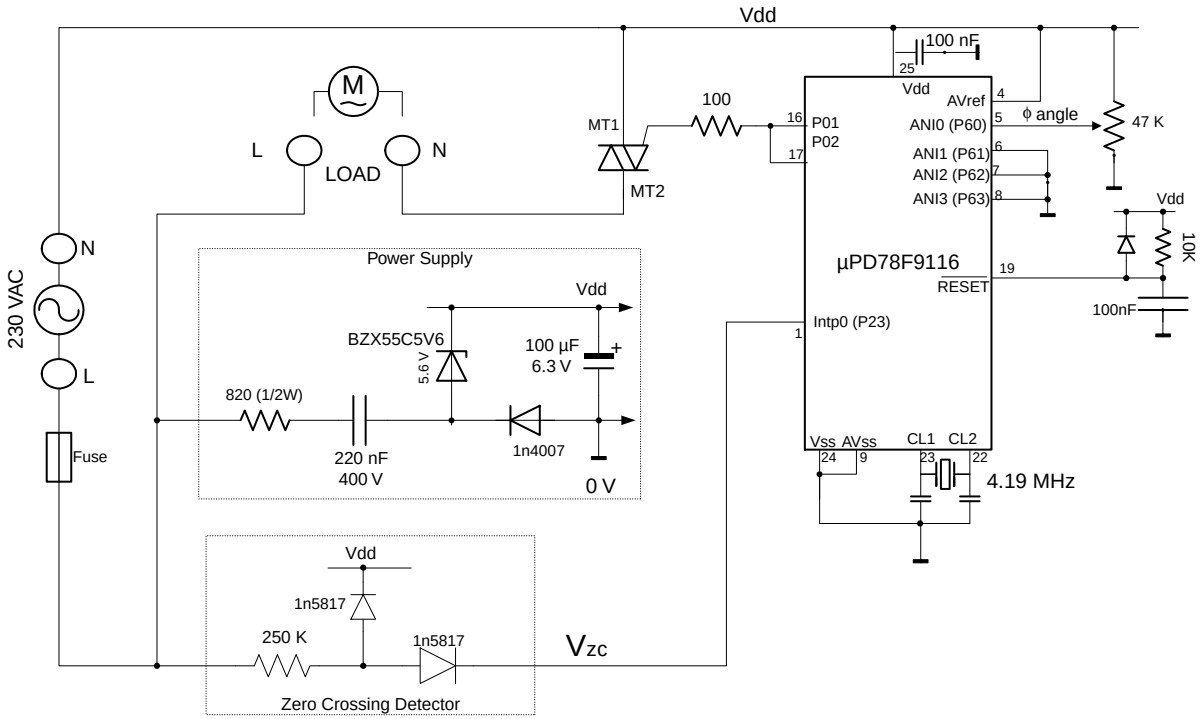
The following shows the schematic diagram for both the µPD78F0103 and µPD78F9116 designs.

Figure 4-1: µPD78F0103 Schematic Diagram



Remark: All other Microcontroller I/O pins are set to Output.

Figure 4-2: µPD78F9116 Schematic Diagram



Remark: All other Microcontroller I/O pins are set to Output.

4.2 Device Configurations

The configuration and operating environment for both low cost members of the 8-bit Microcontroller family, is essentially the same. The exact configurations are shown below.

Table 4-1: Device Configurations

	µPD78F0103	µPD78F9116
Operating clock	4.19 MHz	4.19 MHz
Operating Voltage	5 V	5 V
Internal ROM	24 Kbytes	16 Kbytes
Internal RAM	768 bytes	256 bytes

4.3 Peripherals and I/O Usage

Tables 4-2 and 4-3 below are list all the I/O pins from of the µPD78F0103 and µPD78F9116 devices listed and the ones used in this application are described with there their associated function.

Table 4-2: µPD78F9116 I/O Assignment

Pin No.	Pin Name	Signal Name	Function
1	P23	INTP0	Zero Crossing
2	P24		Not Used
3	P25		Not Used
4	AV_{DD}	+5 V	A/D Reference
5	P60	ANI0	ϕ Control
6	P61		Not Used
7	P62		Not Used
8	P63		Not Used
9	AV_{SS}	0 V	Analogue GND
10	IC0		Not Used
11	P50		Not Used
12	P51		Not Used
13	P52		Not Used
14	P53		Not Used
15	P00	GT1	Gate Trigger
16	P01	GT2	Gate Trigger
17	P02		Not Used
18	P03		Not Used
19	$\overline{RESET}$		Internal POR
20	V_{PP}	V_{PP}	Programming
21	IC0		Not Used
22	X1		Main Crystal
23	X2		Main Crystal
24	V_{SS}	0 V	Normal GND
25	V_{DD}	+5 V	Normal V_{DD}
26	P10		Not Used
27	P11		Not Used
28	P20		Not Used
29	P21		Not Used
30	P22		Not Used

Table 4-3: µPD78F0103 I/O Assignment

Pin No.	Pin Name	Signal Name	Function
1	P33		Not Used
2	P32		Not Used
3	P31		Not Used
4	P30		Not Used
5	V_{PP}	V_{PP}	Programming
6	V_{SS}	0 V	Normal Ground
7	V_{DD}	+5 V	Normal V_{DD}
8	X1		Main Crystal
9	X2		Main Crystal
10	$\overline{\text{RESET}}$	$\overline{\text{RESET}}$	System Reset
11	P03		Not Used
12	P02	GT3	Gate Trigger
13	P01	GT2	Gate Trigger
14	P00	GT1	Gate Trigger
15	P10		Not Used
16	P11		Not Used
17	P12		Not Used
18	P13		Not Used
19	P14		Not Used
20	P15		Not Used
21	P16		Not Used
22	P17		Not Used
23	P130		Not Used
24	P23		Not Used
25	P22		Not Used
26	P21		Not Used
27	P20	ANI0	ϕ Control
28	AV_{REF}	+5 V	A/D reference
29	AV_{SS}	0 V	Analogue Ground
30	P120	INTP0	Zero Crossing

4.4 16-Bit Timer Function

The 16bit timer is only used for synchronisation to the Mains zero crossing.
The operation is slightly different due to the different timer structure between the µPD78F0103 and µPD78F9116. Once the Mains frequency has been determined the 16-bit timer is no longer used and is available for any other user function as required.

Only one 16-bit timer is provided on either of these devices.

- 78K0 - µPD78F0103
  The 16-bit timer is used to detect the Mains frequency. In this example 50 Hz and 60 Hz mains frequencies can be detected. This detection modifies the internal timing etc. of the Phase control sections.
  This approach has been taken due to the selection of input clock frequencies available to the 16-bit timer. This differs from the 78K0 approach

- 78K0S - µPD78F9116
  The 16-bit timer is also used to detect the Mains frequency in a similar way that of the 78K0 example. However, in the 78K0S devices, the 16-bit timer is free running, with a different range of input clock frequencies available. Therefore, the approach to measure the mains frequency is slightly different to that for the 78K0. Here it is convenient to use the 16-bit capture register associated with the timer, as the external Interrupt pin is also the input trigger for the capture register.

4.5 8-Bit Timer Function

The 8-bit timer is used in the same way in both the µPD78F0103 and µPD78F9116, providing the main timing reference for the Phase Angle control function.
The K0 device has a slightly different input clock selection to that of the K0S device, but the timer compare time is set to be the same in both cases. An internal timer interrupt is generated on each compare value match.

The main time is set to 125 µS, which is used to define the pulse width for the Gate trigger pulse. The Phase Angle delay is calculated as multiples of the timer interrupt (i.e. Multiples of 125 µS). The timer is set to Clear and Restart on Compare match interrupt generation.

The timer used in the µPD78F0103 is Timer 50.
The timer used in the µPD78F9116 is Timer 80.

4.6 Watchdog Timer Function

The watchdog in both K0 and K0S devices is used as a safe guard, to avoid program run away and random operation. The watchdog is set to provide an internal Reset if the time-out period expires. This will reset the Triac operation (sets the Gate Trigger outputs inactive (High in this case)) and re-initialises the system.
The Watchdog is set to overflow every 32 mS in both devices.

4.7 A/D converter

The A/D converter in this application is used to provide the Phase Angle delay. Only one pin is used (ANI0), and provides a 10-bit resolution from an external Potentiometer. The configuration is that a zero conversion provides the minimum Phase delay (maximum power), while a Full Scale value is the maximum delay (minimum power).

The A/D converter operates at different conversion rates in the two devices, due to the different pre-scalar clock frequencies available. The settings are as shown below

- µPD78F0103 23 µS (f_X / 2^{96})
- µPD78F9116 14.3 µS ($f_X/2^{60}$)

(f_X = 4.19 MHz)

The minimum conversion time allowed for both A/D converters is 14 µS.
The Analogue reference (AV_{DD}), is connected to the system V_{DD} supply.
Analogue Ground (AV_{SS}), is connected to the Power supply ground reference.

4.8 Interrupt Functions

Two interrupts are used in the application example.

- External Interrupt - INTP0 (Zero crossing Detector)
- 8-bit timer Compare

The external (Zero Crossing) interrupt is used as the main timing reference for the system and provides the main control flow of the software.
The functions serviced by this interrupt are:

- Mains Synchronisation
- Mains Frequency detection
- Positive Zero Crossing Reference
- Negative Zero Crossing Reference

The 8-bit timer compare interrupt is used for two operations
The Timer frequency is fixed in both cases at 125uS (@ 4.19 MHz Crystal frequency).

- The Phase Angle Firing Delay
- The Triac Gate Trigger Pulse Generation

Chapter 5 Software Process Descriptions

5.1 Initialisation

The initialisation process is responsible for the initialising the system after a system reset (or Watchdog Overflow). It configures the basic clock settings of the device, initialises the peripherals that are used for the application and enables the appropriate interrupts. The initialisation contains the following sections as shown below:

(1) The Triac Gate Trigger Port

(2) The 8 and 16 bit Timers

(3) The A/D Converter

(4) Remaining I/O Ports settings

(5) The Watchdog configuration

(6) The Start Up Mode.

5.2 CSTARTUP

The CSTARTUP routine is a modified version of the IAR CSTARTUP program provided as part of the standard C Library

The modification consists of adding the Gate Trigger initialisation code so that the state of the Triac Gate Trigger Pins is defined as quickly as possible. The Code additions consists of setting the Port Mode to Output and setting the Gate Trigger I/O pin to "1".

In order to avoid any problems with the standard IAR C-Library tool installation for future applications, the customised CSTARTUP program was added to the project as a source file. A local copy of the standard C Library file was made and also added to the project and the original IAR CSTARTUP program was removed from the local library.

5.3 A_to_D

The A/D routine is used to define the Phase Angle firing delay, and is sampled at the start of each Positive Zero crossing (i.e. the start of each mains cycle).

The aim of the A/D conversion is that the Phase Angle delay is updated before the first 8-bit timer interrupt occurs. It is important therefore that the time taken for the A/D conversion process does not overrun into the first 8-bit timer interrupt. As the conversion, times are different for the two devices a single conversion is made.

The A/D is started at the beginning of the function and stopped and end of the conversion sequence in order to save power. There is sufficient time for two conversions to take place, the first being ignored, as it may not be accurate. The result is then scaled to provide the Phase Angle delay, with the maximum value being "capped" to ensure that triggering does not occur at the start of or overflow into the next half of the mains cycle. Different maximum values are required for 50 Hz and 60 Hz due to the differing cycle periods.

5.4 Main

The main program includes the initialisation of all the peripherals and system variables. Prior to global interrupts being enabled, the startup "Mode" is set to Mains Synchronisation. (Mode = 2) (see section 5.5).

(1) The clock selection varies from the K0 device and the in both cases the internal clock is set to maximum (f_X = 4.19 MHz).

   (a) In the µPD78F0103, an internal Oscillator (Ring Oscillator) operates immediately upon Power On. This clock is used for the initialisation sequence while the main Crystal is stabilising. Once the main clock is operating, the internal clock is switched over and the Ring Oscillator stopped to save power.

   (b) On the µPD78F9116, there is no internal Ring Oscillator, so the clock is set to maximum as soon as the main crystal and CPU are operating.

(2) The Watchdog configuration for the two devices is also different and requires a slightly different approach to the operation.

   (a) On the K0S device, the watchdog is enabled by setting the "RUN" bit in the control Register. The watchdog settings can be defined prior to this bit being set. In this case, the Watchdog is started once all the initialisation is complete and the Interrupts have been enabled.

   (b) On the K0 device, the watchdog operates immediately on Reset based on the maximum settings. The configuration register can be written to once in order to change the time delay, or even stop the watchdog. In this case, the watchdog configuration and operation is performed after all the other functions have been initialised and enabled.

The Main program operates in an endless loop. The control of the system is provided by the appropriate interrupt routines.

5.5 Zero_Cross (INTP0 - Interrupt Service Routine)

This is main control program in the software. An interrupt is generated for each of the Zero Crossing points. Program control is defined by the "Mode" variable, which defines the operation of the software.

There are four states of operation:

(1) Mode = 0 Positive Zero Crossing Operation

This is the start of each Mains cycle

- The 8 -bit timer is started.
- The A/D conversion process executed.
- The Phase Angle Delay variables are updated.
- The Mode set to the next state (Mode = 1).

(2) Mode = 1 Negative Zero Crossing Operation

This is the start of the 2nd mains half cycle

- The 8-bit timer is started.
- The Mode set to the next state (Mode = 0)

(3) Mode = 2 Mains Synchronization

Simply to synchronise to the start of the mains cycle

- The 16-bit timer is started. (78K0 version only)
- The Mode is set to the next state (mode = 3)

(4) Mode = 3 Mains Frequency Detection

Detection of the applied mains frequency

- 5 positive zero crossing interrupts are measured, the results averaged and the mains frequency detected to be either 50 Hz or 60 Hz
- The Mode is set to main operation (Mode = 0).

5.6 Phase_Firing (8-bit Timer Compare Match ISR)

The Compare Match of the timer is used as the main reference for the Phase Angle firing delay and the Gate Trigger pulse generation.

Basic compensation for the unequal Mark-Space ratio of the zero crossing detection is achieved by the adding an appropriate offset to the Phase angle delay.

The Phase Angle delay value is calculated by the A/D conversion routine, which defines the number of interrupts delay before firing the Triac.

In this application example, we generate four firing pulses to ensure that enough Gate current is supplied to the Triac to ensure the Holding Current is achieved.

Once the delay and firing sequence is complete the timer is stopped.

Chapter 6 Conclusions

These application examples provide an approach to a basic triac control system suitable for implementation with NEC's K0 or K0S 8-bit Microcontroller families.

The examples offer cost-effective solutions for Universal Motor Control applications driven directly from the Microcontroller, while still providing further resource within the Microcontroller for additional user functions or improvements to the control algorithm.

The memory requirements for these examples are as follows

- **78K0S - µPD78F9116**

CODE memory	16 Kbytes available,	918 bytes used
DATA memory	256 bytes available,	74 bytes used

- **78K0 - µPD78F0103**

CODE memory	24 Kbytes available,	792 bytes used
DATA memory	768 bytes available,	74 bytes used

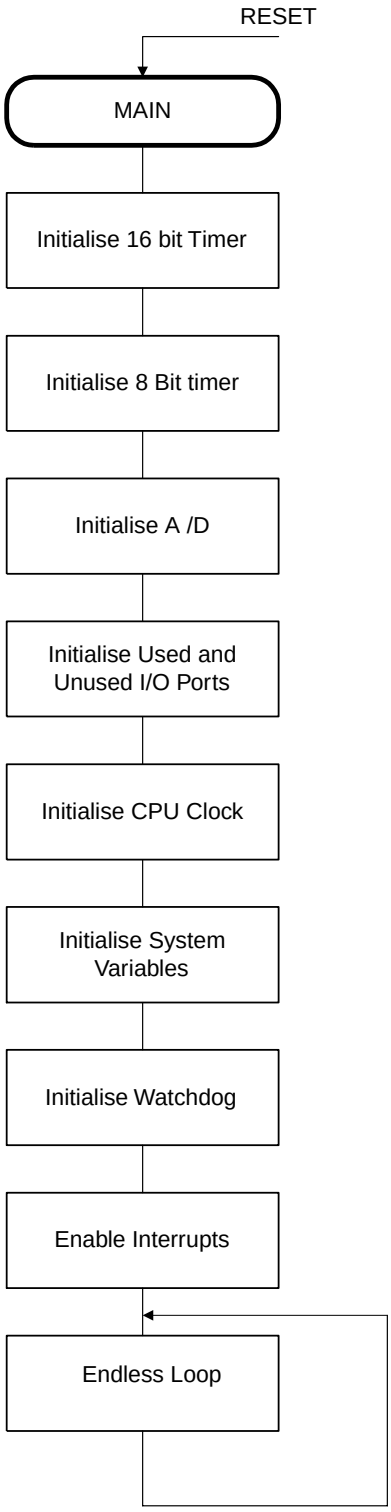
For further details of NEC’s motor control solutions visit the following web site:

http://www.ee.nec.de/motorcontrol

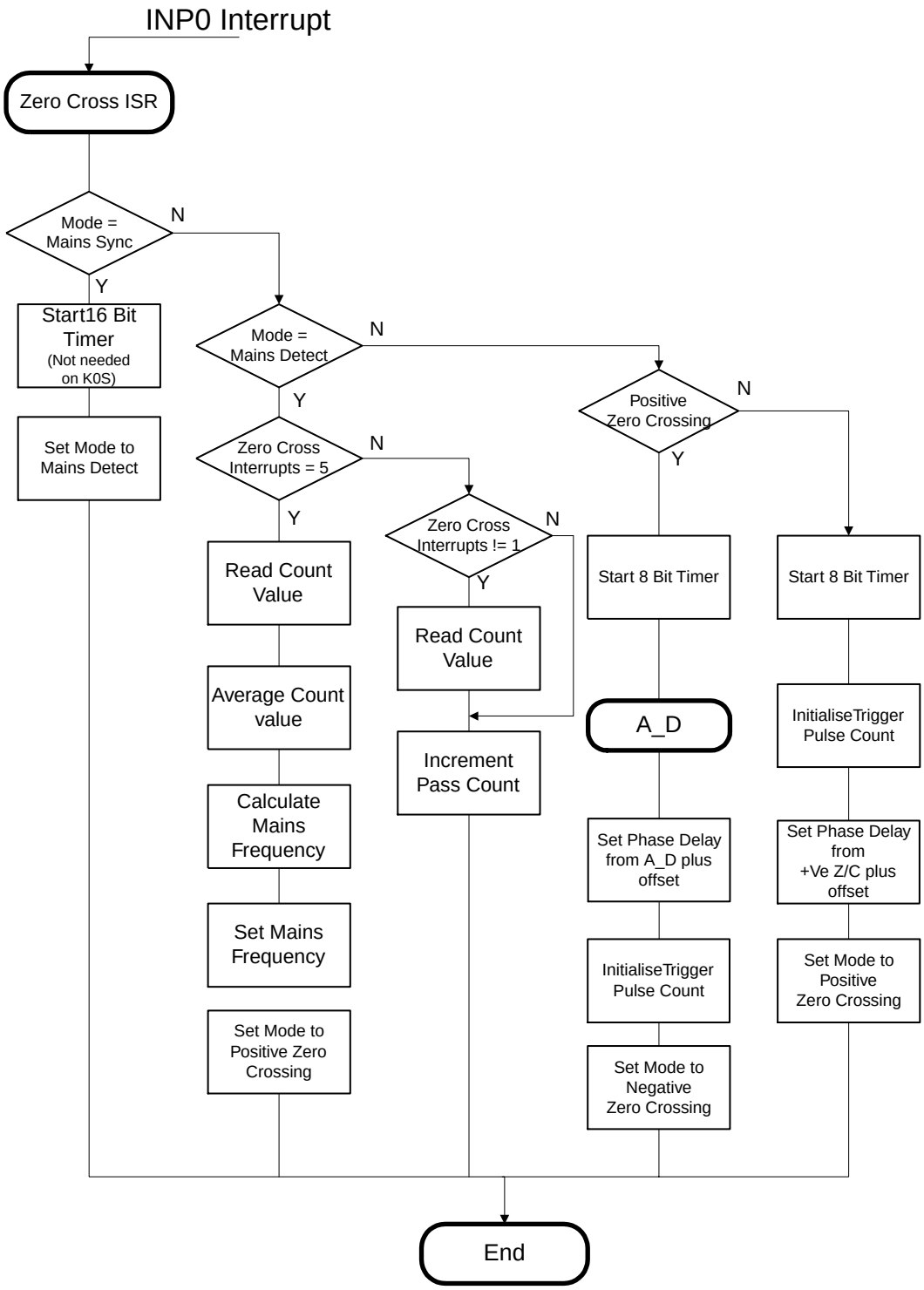
[MEMO]

Chapter 7 Flow-Charts

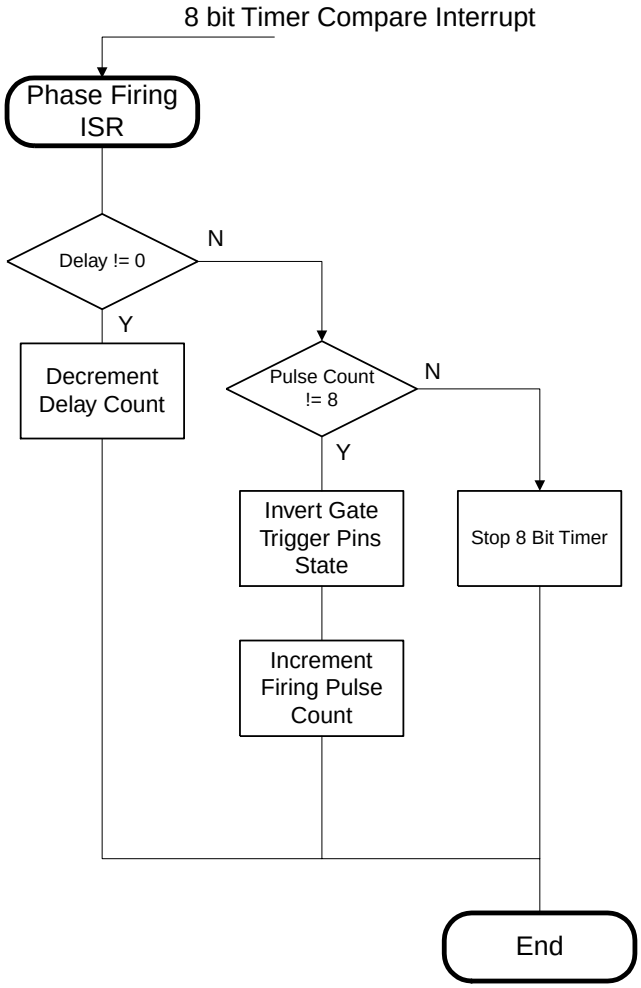
7.1 Main Flow Diagram



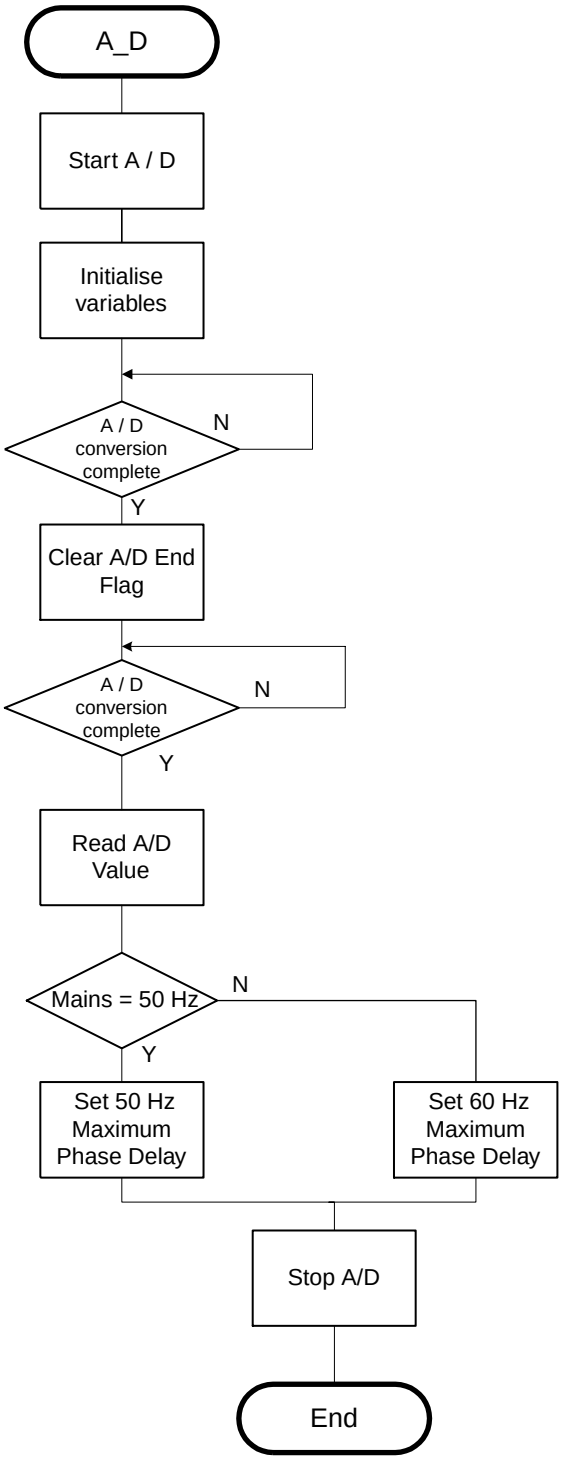
7.2 Zero Cross - ISR



7.3 Phase Firing - ISR



7.4 A_to_D


A_D
Start A / D
Initialise variables
A / D conversion complete
N
Y
Clear A/D End Flag
A / D conversion complete
N
Y
Read A/D Value
Mains = 50 Hz
N
Y
Set 50 Hz Maximum Phase Delay
Set 60 Hz Maximum Phase Delay
Stop A/D
End

Appendix A 78K0 Program Listings

```
/*====================================================================================
** PROJECT        =        Simple Triac Control
** MODULE         =        Main Program
** SHORT DESC.    =        78K0 AC Universal Motor Controller
** VERSION        =        1.0
** DATE           =        4.11.2002
** LAST CHANGE    =
** ====================================================================================
** Description:   Basic Triac control algorithm for Universal AC motor control
**                The Phase Angle Partialisation approach is used and assumes
**                a Three Quadrant Triac.
**                Main Crystal Frequency is Assumed to be 4.19MHz
**                The A/D is used to provide the Phase Angle control
**                The 16 bit Timer is used only for initial mains syncronisation
**                The 8-bit Timer is used for compatibility to other K0 and K0S devices
**                Alternatively an approach without A/D could use the uPD78(F)003x
** ====================================================================================
** Environment:   Device:      uPD78F0103 (78K0 / KB1)
**                (Ring Oscillator assumed to be Operating - Device code option)
**                Assembler:        A78000      Version    V3.33A
**                C-Compiler:       ICC78000    Version    V3.33A
**                Linker:           XLINK       Version    V4.52J
** ====================================================================================
** By:      NEC Electronics Europe GmbH
**          Cygnus House
**          Sunrise Parkway
**          Linford Wood
**          Milton Keynes UK MK14 6NP
** ====================================================================================
Changes:
** ====================================================================================*/

#include <Df0103_Triac_Control.h>
#include <in78000.h>

/* external references */
extern void A_to_D (void);

/* Definitions  */

#define AD_channel     0              // Define A/D channel to use for POT (ANI0)
#define Mains          19000          // 50Hz / 60Hz detection threshold
#define pos_zcross     0              // Mode = Positive zero crossing
#define neg_zcross     1              // Mode = Negative Zero Crossing
#define mains_sync     2              // Mode = Sync to mains (Startup / W-Dog Reset)
#define mains_detect   3              // Mode = Mains frequency detection (50Hz / 60Hz)
#define timer50_compare  130          // 8 bit Timer compare Value (125uS @ 4.19MHz)
#define Pos_ZC_offset  1              // Phase Angle offset from +VE Zero crossing point 1st half cycle
#define Neg_ZC_offset  17             // Phase Angle offset from -Ve Zero crossing point 2nd half cycle
                                              // Compensates for un-equal Zero crossing periods
                                              // Also allows for current Zero crossing offset


/* Variable Declarations  */
```

```
unsigned char Mode;                 // 0 = Normal Control mode, +Ve Mains Cycle
                                            // 1 = Normal Control Mode -Ve Mains Cycle
                                            // 2 = Start Up Sync to Zero Crossing
                                            // 3 = Startup - Detect Mains Frequency


unsigned char count;                // General counter
unsigned char delay;                // Counter used for Phase delay
unsigned short val;                 // General integer
unsigned short Firing_Angle;        // Firing Angle delay (Same for both +/- half cycles
unsigned char mains;                // Indicate 50Hz(0) or 60Hz(1)
unsigned char Gate_Trigger;         // Data to write to port 2 for Triac Gate trigger

/*************************************************************************/


/***********************************
 Zero Crossing Interrupt isr (INTP0)
***********************************/
interrupt [INTP0_vect] void Zero_Cross(void)
{
#ifdef  watchdog_enable
    WDTE = 0xAC;                    // Service Watchdog
#endif

    switch (Mode)                   // Mains sync & +Ve and -Ve mains cycle operation
    {
     case pos_zcross:
          EGN    = 1;                // Enable INTP0 for -Ve edge (Next mains half cycle)
          EGP    = 0;                // Disable INTP0 +Ve edge for extra noise immunity
          TCE50  = 1;                // Start 8 bit Timer
          count  = 0;
          Mode   = neg_zcross;       // Set for -Ve Zero Crossing next
          A_to_D();                  // Check for change in Phase Angle delay (from POT)
                                            /* Result goes into Firing_Angle */
          delay  = Firing_Angle + Pos_ZC_offset;
                                            // Phase angle delay for 1st Half cycle
          break;

     case neg_zcross:
          EGN    = 0;                // Disable INTP0 -Ve edge (-Ve Half cycle. For Noise immunity)
          EGP    = 1;                // Enable INTP0 +Ve edge for start of next mains cycle
          TCE50  = 1;                // Start 8 bit Timer
          count  = 0;                // Set up for -Ve Mains Cycle
          Mode   = pos_zcross;       // Set for +Ve Zero Crossing next

          delay = Firing_Angle + Neg_ZC_offset;
                                            // Phase Angle delay for 2nd half cycle
          break;

     case mains_sync:
          TMC00 = 4;                 // Mains Sync phase 1. start 16 bit timer, Free Running Mode
          Mode = mains_detect;       // Set Startup / Sync 2nd phase (Next +Ve Zero Crossing)
          break;

     case mains_detect:
          if (count != 0)            // Ignore the 1st Zero Crossing
```

```
            {
             val = val + TM00;          // Read new Count & total
             }

            TMC00 = 0;                  // Clear Timer
            TMC00 = 4;                  // Restart Timer
            if (count == 1)
            {
             ++count;
             }

            else if (count !=5)
            {
             ++count;                   // increment mains sync pass count (5 total)
             val = val/2;               // running average
             }
             else
             {
              Mode  = 0;                // sync done, Normal mode
              TMC00  = 0;               // Stop 16 bit timer
              val = val /2;

              if (val > Mains)
              {
                mains = 0;              // 50Hz
               }
               else
               {
                 mains = 1;             // 60Hz
                }
               }
            break;

      default:  break;
     }
}


/****************************
      8 bit Timer(50) isr
****************************/

interrupt [INTTM50_vect] void Phase_firing(void)
{
            if (delay !=0)              // Wait for Triac Gate firing point
            {
             --delay;
             }
       else if (count !=8)
             {
              ++count;                  // More Trigger pulses (4 periods total)
              Gate_Trigger ^= 7;                // Invert Triac Gate Outputs (P26 & P27)
              P0  = Gate_Trigger;               // Output Triac Gate Trigger
              }                         /* 3 Outputs used to ensure Trigger current.
                                           Pins MUST be set together */
       else
       {
        TCE50  = 0;                     // stop 8 bit timer
```

```
        }
}


/*****************************
 Main C Program Entry point
*****************************/
void main (void)
{

/**********************
 System Initialisation
**********************/


/*  16 Bit Timer    -   Only used for mains syncronisation */

      TMC00   = 0;                  // Stop 16 bit Timer operation
      TOC00   = 0;                  // No output
      TMMK010 = 1;                  // Disable 16 bit Timer Interrupts
      TMMK000 = 1;
      PRM00   = 1;                  // 16 bit timer Prescaler set to fx / 2^2



/* Timer 50     -   Other 8 bit Timers Not used in this application */

      TMC50   = 0;                  // Stop 8 bit timer(50)& define Setup
                                          /* Clear on Match on CR50, All other modes disabled */
      TMMK50  = 0;                  // Ensure Timer 50 Interrupt Masked
      TCL50   = 4;                  // Clock set to fx / 2^2 = 0.95uS @ 4.19MHz
      CR50    = timer50_compare;            // Initial Compare value set to 125uS period


/* A/D      -   A/D Used for Firing Angle control */

      ADS     = AD_channel;         // Set "POT" input channel
      PFM     = 0;                  // ensure Power fail operation disabled
      ADMK    = 1;                  // Mask A/D Interrupt. A/D "done" will be polled
      ADIF    = 0;                  // Clear Interrupt flag (i.e A/D Done)
      ADM     = 0x30;               // fx / 96 -> 23uS @ 4.19MHz, A/D Stopped, Boost Off


/* Used Ports */
/* Note - Port 0 (P00, P01 & P02, Triac Gate) are defined in assembler file Cstartup.s26   */

     EGP   = 1;                     // Enable INTP0 to +Ve edge Interrupt for syncronisation
     EGN   = 0;                     // Disable INTP0 -Ve edge interrupt during Syncronisation
     PIF0  = 0;                     // Clear INTP0 pending flag (Wait for next Z/C)
     PMK0  = 0;                     // enable external interrupt INTP0 (Z/C)

/* Unused Ports   -   Set to Output for Additional Noise Immunity  */

    PM1   = 0;
    PM3   = 0;
    P1    = 0;
    P3    = 0;
```

```
/* Set Clock speed (fx)  */

      PCC   = 0;                    // Set Clock prescaler to Max (4.19Mhz)
      _ASM("NOP");                  // Settling delay
      _ASM("NOP");

      MCM0  = 1;                    // Select Main Crystal
      while (MCS != 1);             // Ensure Crystal to operating
      RSTOP = 1;                    // Stop Ring Oscillator



/*******************************
  main program starts here
*******************************/

      Mode    = mains_sync;         // Set Startup mode - sync to Mains
      Gate_Trigger = 7;             // Set Gate Output Buffer
      count   = 0;
      _EI();                                // Enable Global Interrupts

/* Start or Stop Watchdog - Used for safety (In 78K0/KB1 Watchdog starts on Reset ) */

#ifdef watchdog_enable              /* Use Watchdog */

      WDTM  = 0x6C;                 // fx / 2^17, 31mS @ 4.19MHz, Main Clock, Watchdog Run

#else                                       /* Stop Watchdog Debugging Mode */

      WDTM  = 0x7C;                 // Watchdog Stopped

#endif

      while (1);                            // Wait for interrupts
}



/*============================================================================
** PROJECT        =        Simple Triac Control
** MODULE         =        A to D Conversion Program
** SHORT DESC.  =          78K0 Triac Control Example
** VERSION        =        1.0
** DATE           =        4.11.2002
** LAST CHANGE  =
** ============================================================================
** Description:     The program provides a basic Triac control algorithm for
**                  Universal AC motor control. The design is based on the
**                  Phase Angle Partialisation approach and assumes
**                  a Three Quadrant Triac used.
**                  Crystal Frequency is Assumed to be 4.19MHz
**                  The A/D is used to provide the Phase Angle control
**                  Alternatively an approach without A/D could use the uPD78(F)003x
** ============================================================================
** Environment:     Device:             uPD78F0103 (78K0 / KB1)
**                   (Ring Oscillator assumed to be Operating - Device code option)
```

```
**                  Assembler:          A78000      Version    V3.33A
**                  C-Compiler:         ICC78000    Version    V3.33A
**                  Linker:             XLINK       Version    V4.52J
** ============================================================================
** By:      NEC Electronics (UK) Ltd
**          Cygnus House
**          Sunrise Parkway
**          Linford Wood
**          Milton Keynes UK MK14 6NP
** ============================================================================
Changes:
** ============================================================================*/

#include <Df0103_Triac_Control.h>

extern unsigned short Firing_Angle;             // Phase Angle
extern unsigned char mains;                     // 50Hz or 60Hz flag

/*****************************************************************************/

void A_to_D (void)
{
    ADIF = 0;                                   // Clear A/D End flag
    ADCS  = 1;                                          // Start A/D
    while (ADIF !=1);                                   // wait for next A/D conversion end
    ADIF = 0;                                   // Ignore 1st Conversion
    while (ADIF !=1);                                   // wait for next A/D conversion end

    Firing_Angle =(ADCR >> 10);                 // Convert & Scale A/D value

    if (Firing_Angle > 50)                              // 50Hz - Check for Maximum delay
    {
     Firing_Angle = 50;                                 // Ensure Maximum delay value < half period
     }

    if (mains != 0 & Firing_Angle > 40)         // If 60Hz then adjust maximum delay value
    {
     Firing_Angle = 40;
     }
    ADCS = 0;                                           // Stop A/D
}
```

Appendix B 78K0S Program Listings

```
/*==========================================================================
** PROJECT       =       Simple Triac Control
** MODULE        =       Main Program
** SHORT DESC.   =       78K0S AC Universal Motor Controller
** VERSION       =       1.0
** DATE          =       4.11.2002
** LAST CHANGE =
** ==========================================================================
** Description:    Basic Triac control algorithm for Universal AC motor control
**                 The Phase Angle Partialisation approach is used and assumes
**                 a Three Quadrant Triac.
**                 The main crystal frequency is assumed to be 4.19MHz
**                 The A/D is used to provide the Phase Angle control
**                 The 16 bit Timer is used only for initial mains syncronisation
**                 The 8 bit Timer is used for compatibility to other K0 and K0S devices
**                 Alternatively an approach without A/D could use the uPD78(F)907x
** ==========================================================================
** Environment:    Device:         uPD78F9116
**                 Assembler:      A78000          Version   V3.33A
**                 C-Compiler:     ICC78000        Version   V3.33A
**                 Linker:         XLINK           Version   V4.52J
** ==========================================================================
** By:     NEC Electronics Europe GmbH
**         Cygnus House
**         Sunrise Parkway
**         Linford Wood
**         Milton Keynes UK MK14 6NP
** ==========================================================================
Changes:
** ==========================================================================*/

#include <Df9116A_triac_control.h>
#include <in78000.h>

/* external references */
extern void A_to_D (void);

/* Definitions  */

#define AD_channel     0          // Define A/D channel to use for POT (ANI0)
#define Mains          1500       // 50Hz / 60Hz detection threshold
#define pos_zcross     0          // Mode = Positive zero crossing
#define neg_zcross     1          // Mode = Negative Zero Crossing
#define mains_sync     2          // Mode = Sync to mains (Startup / W-Dog Reset)
#define mains_detect   3          // Mode = Mains frequency detection (50Hz / 60Hz)
#define timer80_compare  65       // 8 bit Timer compare Value (125uS @ 4.19MHz)
#define Pos_ZC_offset  1          // Phase Angle offset from +VE Zero crossing point 1st half cycle
#define Neg_ZC_offset  16         // Phase Angle offset from -Ve Zero crossing point 2nd half cycle
                                        // Compensates for un-equal Zero crossing periods
                                        // Also allows for current Zero crossing offset


/* Variable Delcalarations  */
```

```
unsigned char Mode;                 // 0 = Normal Control mode, +Ve Mains Cycle
                                            // 1 = Normal Control Mode -Ve Mains Cycle
                                            // 2 = Start Up Sync to Zero Crossing
                                            // 3 = Startup - Detect Mains Frequency
unsigned char count;                // General counter
unsigned char delay;                // Counter used for Phase delay
unsigned short val;                 // General integer
unsigned short Firing_Angle;        // Firing Angle delay (Same for both +/- half cycles
unsigned char mains;                // Indicate 50Hz(0) or 60Hz(1)
unsigned char Gate_Trigger;         // Data to write to port 2 for Triac Gate trigger

/*****************************************************************************/


/***********************************
 Zero Crossing Interrupt isr (INTP0)
***********************************/
interrupt [INTP0_vect] void Zero_Cross(void)
{
#ifdef watchdog_enable
    RUN   = 1;                      // Service Watchdog
#endif

    switch (Mode)                   // Mains sync & +Ve and -Ve mains cycle operation
    {
     case pos_zcross:
          INTM0  = 0;               // Set INTP0 for -Ve edge (Next mains half cycle)
          TCE80  = 1;               // Start 8 bit Timer
          count  = 0;
          Mode   = neg_zcross;      // Set for -Ve Zero Crossing next
          A_to_D( );                // Check for change in Phase Angle delay (from POT)

          delay  = Firing_Angle + Pos_ZC_offset;
                                            // Phase angle delay for 1st Half cycle
          break;

     case neg_zcross:
          INTM0  = 4;               // Set INTP0 +Ve edge (Next Mains Half cycle)
          TCE80  = 1;               // Start 8 bit Timer
          count  = 0;               // Set up for -Ve Mains Cycle
          Mode   = pos_zcross;      // Set for +Ve Zero Crossing next

          delay = Firing_Angle + Neg_ZC_offset;
                                            // Phase Angle delay for 2nd half cycle
          break;

     case mains_sync:
          Mode = mains_detect;      // Set Startup / Sync 2nd phase (Next +Ve Zero Crossing)
          break;

     case mains_detect:
          if (count == 0)           // 1st Zero Crossing measurement point
          {
           val = TCP20;             // Yes ! Read initial capture value
           TOF20 = 0;               // Clear Overflow Flag
           }
```

```
            else
            {
             if (TOF20 == 0)                // No - Timer Overflow ?
             {
              val = TCP20 - val;            // No - Calc & Store new value
             }
              else
              {
               val  = (65536 - val) + TCP20;
                                             // If Yes, Add overflow & calc new value
               TOF20 = 0;                    // Clear Overflow flag
               }
              Firing_Angle = Firing_Angle + val;       // Sum count
                                             /* (Firing Angle used as General variable here */

              val  = TCP20;                  // Re Load capture value for next calculation
              }

            if (count !=5)
            {
             ++count;                        // increment mains sync pass count (5 total)
            }
             else
             {
             Firing_Angle = Firing_Angle / 4;
                                             // average sync count values
             Mode  = pos_zcross;             // sync done, Normal mode (+Ve Zero Crossing)
             TMC20 = 0;                      // Disable 16 bit Timer Capture

             if (Firing_Angle > Mains)
             {
              mains = 0;                     // 50Hz
              }
              else
              {
               mains = 1;                    // 60Hz
              }
              }
            break;

       default:  break;
      }
}


/*****************************
      8 bit Timer(80) isr
*****************************/

interrupt [INTTM80_vect] void Phase_firing(void)
{
            if (delay !=0)                   // Wait for Triac Gate firing point
            {
             --delay;
             }
       else if (count !=8)
             {
              ++count;                       // More Trigger pulses (4 periods total)
```

```
            Gate_Trigger ^= 6;                  // Invert Triac Gate Outputs (P01 & P02)
            P0  = Gate_Trigger;                 // Output Triac Gate Trigger
           }                                    /* 2 Outputs used to ensure Trigger current
                                                   Pins MUST be set together */
       else
       {
        TCE80  = 0;                             // stop 8 bit timer
        }
}


/*****************************
 Main C Program Entry point
*****************************/
void main (void)
{

/**********************
 System Initialisation
**********************/

/* Set Clock speed (fx)  */

      PCC   = 0;                                // Set Clock prescaler to Max (4.19Mhz)
      _ASM("NOP");                              // Settling delay
      _ASM("NOP");


/*  16 Bit Timer    -   Only used for mains syncronisation */

      TMC20   = 0x12;                           // 16 bit Timer Set Up
                                                        /* fx / 2^6, Capture on +Ve edge, No Output */
      TMMK20  = 1;                              // Disable 16 bit Timer Interrupt


/* Timer 80  Used for Basic Timing & Triac Trigger Pulses */

      TMC80   = 2;                              // Stop 8 bit timer(80) & define timer setup
                                                        /* fx / 2^3 (1.9uS @ 4.19MHz),Counter Mode, No Output */
      TMMK80  = 0;                              // Timer 80 Interrupt Enabled
      CR80    = timer80_compare;                        // Initial Compare value set to 125uS period


/* A/D     -   A/D Used for Firing Angle control */

      ADM0    = 0x28;                           // fx / 60 -> 14.3uS @ 4.19MHz, A/D Stopped
      ADS0    = AD_channel;                     // Define "POT" input channel
      ADMK0   = 1;                              // Mask A/D Interrupt. A/D "done" will be polled


/* Watchdog    -   Used for safety */

     TCL2  = 6;                                 // fx / 2^17, 32mS @ 4.19MHz
     WDTM  = 0x18;                              // Watchdog set to Reset Mode


/* Used Ports */
```

```
/* Note - Port 2 (P26 & P27, Triac Gate) are defined in assembler file Cstartup.s26   */

    INTM0 = 1;                  // Enable INTP0 to +Ve edge Interrupt for syncronisation
    PIF0  = 0;                  // Clear INTP0 pending flag (Wait for next Z/C)
    PMK0  = 0;                  // enable external interrupt INTP0 (Z/C)

/* Unused Ports - Set to Output if possible */

    PM2 = 8;                    // All Output except Intp0
    P2  = 0;
    PM5 = 0;
    P5  = 0;
    PM1 = 0;
    P1  = 0;

/*******************************
  main program starts here
*******************************/

    Mode    = mains_sync;       // Set Startup mode - sync to Mains
    Gate_Trigger = 6;           // Set gate trigger O/P buffer
    count   = 0;
    _EI();                              // Enable Global Interrupts

#ifdef watchdog_enable          /* Use / Disable Watchdog as  required for debug  */
    RUN     = 1;                // Start Watchdog
#endif

    while (1);                          // Wait for interrupts
}


/*============================================================================
** PROJECT       =       Simple Triac Control
** MODULE        =       A to D Conversion Program
** SHORT DESC.   =       78K0S AC Universal Motor Controller
** VERSION       =       1.0
** DATE          =       4.11.2002
** LAST CHANGE =
** ==========================================================================
** Description:  The program provides a basic Triac control algorithm for
**               Universal AC motor control. The design is based on the
**               Phase Angle Partialisation approach and assumes
**               a Three Quadrant Triac used.
**               The main crystal frequency is assumed to be 4.19MHz
**               The A/D is used to provide the Phase Angle control
**               Alternatively an approach without A/D could use the uPD78(F)907x
** ==========================================================================
** Environment:  Device:       uPD78F9116
**               Assembler:    A78000     Version   V3.33A
**               C-Compiler:   ICC78000   Version   V3.33A
**               Linker:       XLINK      Version   V4.52J
** ==========================================================================
** By:           NEC Electronics (UK) Ltd.
**               Cygnus House
**               Sunrise Parkway
**               Linford Wood
**               Milton Keynes UK MK14 6NP
```

```
** ============================================================================
Changes:
** ============================================================================*/

#include <Df9116A_triac_control.h>

extern unsigned short Firing_Angle;          // Phase Angle
extern unsigned char mains;                  // 50Hz or 60Hz flag

/*****************************************************************************/

void A_to_D (void)
{
   ADIF0   = 0;
   ADCS0   = 1;                              // Start A/D

   while (ADIF0 !=1);                                // wait for A/D conversion end

   ADIF0   = 0;                              // Ignore 1st Conversion
   while (ADIF0 !=1);                                // wait for A/D conversion end

  Firing_Angle  = (ADCR0 >> 10);             // Read A/D result & calculate delay
                                                     /* Divisor = 16 Sub periods */

  if (Firing_Angle > 50)                     // 50Hz - Check for Maximum delay
  {
   Firing_Angle = 50;                                // Ensure Maximum delay value < half period
   }

  if (mains != 0 & Firing_Angle > 40)        // If 60Hz then adjust maximum delay value
  {
   Firing_Angle = 40;
   }
  ADCS0  = 0;                                // Stop A/D
}
```

NEC

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

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

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

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

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

Asian Nations except Philippines
NEC Electronics Singapore Pte. Ltd.
Fax: +65-6250-3583

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

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	☐	☐	☐	☐
Technical Accuracy	☐	☐	☐	☐
Organization	☐	☐	☐	☐

[MEMO]