Table of Contents

Part I Overview 4

Part II User Guide 4

1 Hex Editor 4

2 Hex-Dec Convertor 5

3 Keyboard Shortcuts 5

Part III UUSP (UPA-USB Serial Programmer) 6

1 Supported Devices 6

2 Jumpers and Connectors Description 7

3 Atmel 8051, AVR 8-Bit Risk 8

4 Microchip PICs 8

5 EEPROMs 9

I2C and SPI 9

Microwire 9

M35080 10

SDA(E)2506 10

6 Motorola HC05 11

MC68HC05B PLCC52 11

MC68HC705B16 PLCC52 12

MC68HC05H12 PLCC52 13

MC68HC05L28 PDIP56 14

MC68HC05X16/32 QFP64 15

7 Motorola HC08 16

MC68HC(9)08 QFP64 16

MC68HC08AZ32 QFP100 17

MC68HC08AS20 PLCC52 18

8 Motorola HC11 19

MC68HC11A8/E9 PLCC52 19

MC68HC11E QFP64 20

MC68HC11E SDIP56 21

MC68HC11EA9 PLCC52 22

MC68HC11F1 PLCC68 23

MC68HC11K PLCC84 24

MC68HC11K QFP80 25

MC68HC11KA2/4 PLCC68 26

MC68HC11KA2/4 QFP64 27

MC68HC11KG4 QFP100 28

MC68HC11KS PLCC68 29

MC68HC11L6 PLCC68 30

MC68HC11P2 PLCC84 31

MC68HC11PA8 QFP64 32

MC68HC11PH8 PLCC84 33

9 Motorola HC12 34

MC68HC(9)12B32 QFP80 34
MC68HC(9)12D60(A) QFP80 35
MC68HC(9)12D60(A)/DG128(A) QFP112 36
10 Motorola HCS12 37
MC9S12Dx64/128/256 QFP80 37
MC9S12Dx64/128/256 QFP112 38
11 78K0/HC912 Adapter 39
Jumpers and Connectors Description 40
uPD780824/6/8A 41
uPD780973/4 42
uPD780948/9 43
68HC912 QFP112 43
12 STMicroelectronics ST6 45
ST6240 QFP80 45
ST6245 QFP52 46
ST6249 QFP68 47
13 TMS Adapter 48
Socket Description 48
Additional Adapter Schematics 49
TMS370cx36 Adapter Schematic 49
TMS370cx42 Adapter Schematic 49
Part IV Pascal Script Reference 50
1 Device Management 50
AddAction 50
AddDevice 50
AddDeviceGroup 50
BlankCheckDevice 50
GetDevice 50
HideDeviceOrGroup 50
ProgramDevice 50
ReadDevice 50
ShowDeviceOrGroup 50
VerifyDevice 50
2 File I/O 51
AddOpenFileAction 51
OpenFile 51
3 Hex Editor 51
GetByteHexEdit 51
GetSizeHexEdit 51
RefreshHexEdit 51
SelectAllMemoryRange 51
SelectEEPROMRange 51
SetByteHexEdit 51
SetProgramModifiedOnly 51
SetProgramRange 51
4 Message and Input Boxes 52
AddMsg 52
ClearMsg 52
InBox 52
MsgBox 52

5 Miscellaneous 53

Application 54

InputForm 54

IntToHex 54

SetProductInfo 54

6 RemObjects Pascal Script 54

Library 55

Reserved words 55

Statements 56

Types 57

Index 0

1 Overview

Features

Hex Editor
Over write or insert mode
Support hexadecimal, decimal, octal and binary systems
File size up to 2GB (depends on the virtual memory of the computer)
Grouping bytes
Print the whole file or selected part of it
Unlimited Undo/Redo
Ajustable bytes per line
Fast searching/replacing hex or text data
Compare files
Font and colour options
Opening/Saving Intel Hex Format files
Opening Motorola S Record files
Swap even and odd bytes
Copy dump to clipboard
Copy part of a file to another file or to a text editor
Go to specified offset
Fill a selected part of the file in 0 or 255 (FFh)

2 User Guide

2.1 Hex Editor

The hexadecimal editor (HexEdit) allows customer to edited binary files, for programming a memory or micro controller. Maximal size of the file is theoretical 2 GB, but actually depends on available virtual memory of the computer. Editor works in overwriting or insert mode, switched by Insert key or by the button Insert/Over located on the bottom of the window status bar. Hex Edit allows a few files to be opened and various operations to be done with them. Hex Edit has 3 areas: offset, numerical and text.

```
000000: 61 62 73 64 65 66 67 68 absdefgh
000008: 6A 6B 6C 6D 6E 6B 70 71 jklmnkpq
```

Status Bar
Status bar displays the offset of the pointer position from the beginning of the file, the current value located at this offset and the size of the file. There are few buttons available:
Offset button - Toggles hexadecimal, decimal or octal representing of the offset
Data button - Toggles hexadecimal, decimal, octal or binary representing of the numbers
Size button - Toggles hexadecimal, decimal or octal representing of the file size
Find/Replace button - Show/Hide Find/Replace Bar
Monitor button - Show/Hide Data Monitor Bar
Two editing controls allows translating the pointer position at specified offset and data editing (Press Enter in the end)
Find/Replace Bar
This bar allows searching/replacing of text or hexadecimal number forward or backward. Text searching is not case sensitive. If a case sensitive searching is required, click Text button to convert entered text to ASCII codes. Hexadecimal searching is always case sensitive.
Monitor Bar
There are two buttons on the Monitor Bar. First one specifies the size of the number- 8, 16, 32 or 64 bits. The second button changes the order of the bytes - Intel (less signed byte first); Motorola (most signed byte first). The number are displayed as unsigned integer, signed integer and a real number
Working with Clipboard

Hex edit clipboard to copy numbers or text from one file to another one. It's possible copying from UPA to a text editor (Notepad, Word). In this case, the caret position specifies the form of the copied data.

The caret is located at number area

```
24 07 F0 71 7B 51 A1 66 -
```

The caret is located in the text area

```
$.pq{QЎf
```

A Dump can be copied by Edit/Copy as Text

```
005FF8: 036 007 240 113 123 081 161 102 $.pq{QЎf
006000: 000 034 161 120 000 036 161 004 ."Ўx.$Ў.
006008: 000 038 161 015 000 040 239 149 .&Ў..(п
```

It's possible to copy text from a text editor to UPA's hex editor

```
000000: 49 74 27 73 20 70 6F 73  It's pos
000008: 73 69 62 6C 65 20 74 6F  sible to
000010: 20 63 6F 70 79 20 74 65   copy te
000018: 78 74 20 66 72 6F 6D 20  xt from
000020: 61 20 74 65 78 74 20 65  a text e
000028: 64 69 74 6F 72 20 74 6F  ditor to
000030: 20 55 50 41 27 73 20 68   UPA's h
000038: 65 78 20 65 64 69 74 6F  ex edito
```

Keyboard Shortcuts

Left, Right, Up, Down	Moves the caret
End	Moves the caret to the end of the line
Home	Moves the caret to the start of the line
CTRL+End	Moves caret to the end of the file
CTRL+Home	Moves caret to the start of the file
Tab	Toggles between hex and text area
PgDn	Moves the caret down by one page
PgUp	Moves the caret up by one page
Shift+Arrow keys, Home,End, PgDn, PgUp	Selects an area
Ins	Toggles between Insert and Over write modes
Ctrl+Ins, Ctrl+C	Copy
Shift+Ins, Ctrl+V	Paste
Ctrl+X	Cut
Backspace, Del	Delete
Ctrl+Z	Undo
Ctrl+Y	Redo

2.2 Hex-Dec Convertor

Using this option the customer converts numbers from hexadecimal to decimal system and opposite. The type of the number can be choose by a button (on the second line)

2.3 Keyboard Shortcuts

Hex Editor

Left, Right, Up, Down	Moves the caret
End	Moves the caret to the end of the line

Home	Moves the caret to the start of the line
CTRL+End	Moves caret to the end of the file
CTRL+Home	Moves caret to the start of the file
Tab	Toggles between hex and text area
PgDn	Moves the caret down by one page
PgUp	Moves the caret up by one page
Shift+Arrow keys, Home,End, PgDn, PgUp	Selects an area
Ins	Toggles between Insert and Over write modes
Ctrl+Ins, Ctrl+C	Copy
Shift+Ins, Ctrl+V	Paste
Ctrl+X	Cut
Backspace, Del	Delete
Ctrl+Z	Undo
Ctrl+Y	Redo

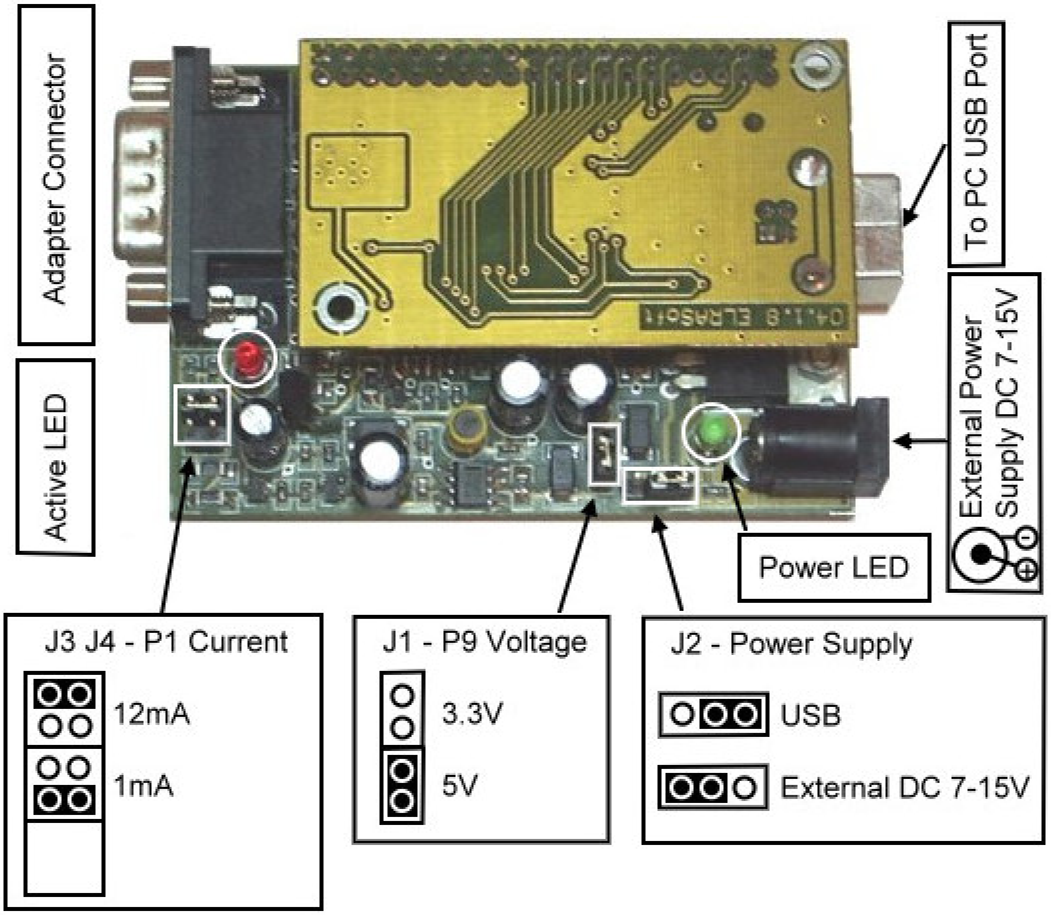
3 UUSP (UPA-USB Serial Programmer)

3.1 Supported Devices

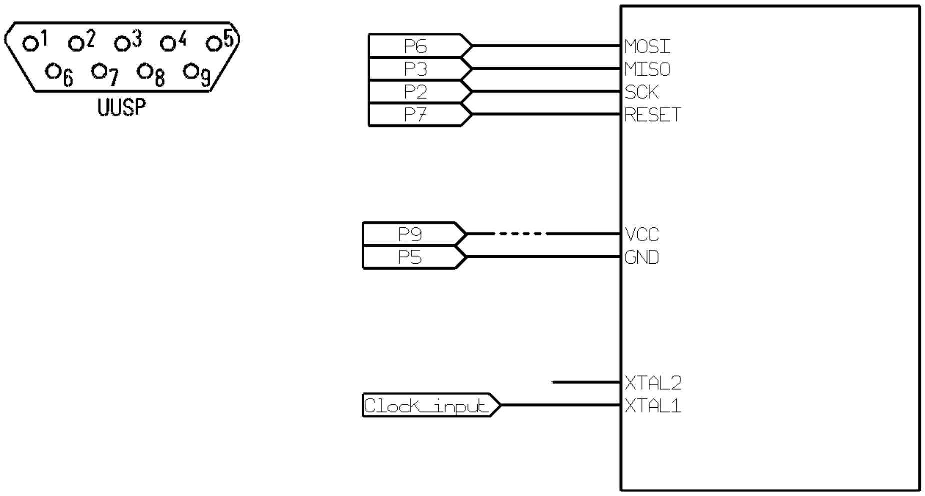
NSC*: CR16HCS5/9, CR16MCS5/9, CR16MES5/9, CR16MFS5/9, CR16MCT5/9, CR16HCT5/9
Motorola HC05*: MC68HC05B6, MC68HC05B8, MC68HC05B16, MC68HC705B16, MC68HC05B32, MC68HC05E6, MC68HC705E6, MC68HC05H12, MC68HC05L28, MC68HC05P3, MC68HC705P3*,MC68HC05X16, MC68HC05X32
Motorola HC08*: MC68HC08AS20, MC68HC08AS32, MC68HC08AS60, MC68HC08AZ32, MC68HC(9)08AZ32A, MC68HC908AZ60, MC68HC908AZ60A
Motorola HC11*: MC68HC11A1, MC68HC11A8, MC68HC11E9, MC68HC11EA9, MC68HC11E20, MC68HC11F1, MC68HC11K4, MC68HC11KA2, MC68HC11KA4, MC68HC11KG4, MC68HC11KS2, MC68HC11KS8, MC68HC11L6, MC68HC11P2, MC68HC11PA8, MC68HC11PH8
Motorola HC12*: MC68HC912B32, MC68HC912BE32, MC68HC912D60, MC68HC912D60A, MC68HC912DC128A, MC68HC912DG128, MC68HC912DG128A
Motorola HCS12*: MC9S12D64, MC9S12A128, MC9S12DG128, MC9S12DG256, MC9S12H128, MC9S12H256
Atmel 8051 Architecture: AT89S51, AT89S52, AT89S53, AT89S8252, AT89S8253
Atmel AVR 8-Bit Risk: AT90S1200, AT90S2313, AT90S2323, AT90S2333, AT90S2343, AT90S4433, AT90S4434, AT90S8515, AT90S8535, ATmega8, ATmega16, ATmega161, ATmega162, ATmega163, ATmega323, ATmega64, ATmega103, ATmega128, ATtiny12, ATtiny15, ATtiny2313, ATmega8515, ATmega8535
Microchip PIC12: PIC12F508, PIC12F509, PIC12F629, PIC12F675
Microchip PIC16: PIC16F627(A), PIC16F628(A), PIC16F648A, PIC16F72, PIC16F73, PIC16F74, PIC16F76, PIC16F77, PIC16F818, PIC16F819
PIC16F83, PIC16F84(A), PIC16F870, PIC16F871, PIC16F872, PIC16F873(A), PIC16F874(A), PIC16F876(A), PIC16F877(A)
EEPROMs I2C: 24C01, 24C02, 24C04, 24C08, 24C16, 24C32, 24C64, 24C65, 24C128, 24C256, 24C512, 85C72, 85C82, 85C92, BAW574252, GRM-003, GRM-004, GRM-005, KKZ-06F, MCM2814, PCA8581, PCF8581, PCF8582, PCF8594, PCF8598, PCF85102, PCF85116, SDA2516, SDA2526, SDA2546, X24C00, X24C01
EEPROMs Microwire: 7002, 93C06, 93C14, 93C46, 93C56, 93C57, 93C66, 93C76, 93C86, 93S46,

93S56, 93S66, GRN-001, GRO-002, KKZ-01, S220, S2914, ST61907, XLS93C46
EEPROMs SPI: M35080, 25C010, 25C020, 25C040, 25C080, 25C128, 25C160, 25C256, 25C320, 25C640, M25P05, M25P10, M25P20, M25P40, M25P80, ST95010, ST95020, ST95040, ST95080, ST95160, ST95320, ST95640, ST95P02, ST95P04, ST95P08, X5043, X5045
EEPROMs Miscellaneous: CXK1011, CXK1012, CXK1013, M6M80011, M6M80021, M6M80041, SDE2506, TC89101, TC89102, 77005, 77007, BR9010, BR9020, BR9040, CAT64LC10, CAT64LC20, CAT64LC40
*EEPROM Only

3.2 Jumpers and Connectors Description

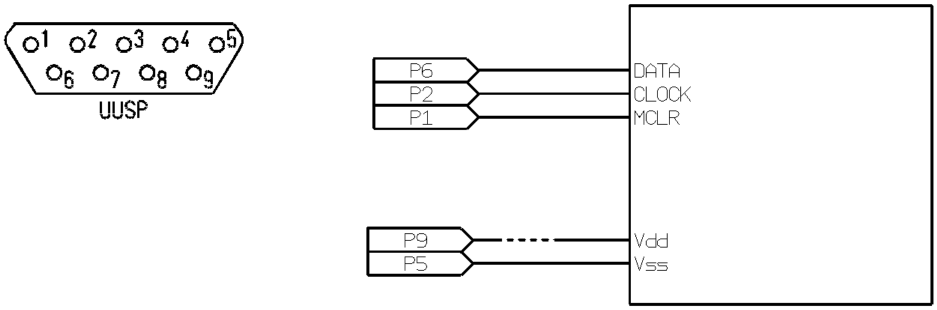


3.3 Atmel 8051, AVR 8-Bit Risk



Either an external clock is supplied at pin XTAL1
or a crystal needs to be connected across pins XTAL1 and XTAL2
Programming of Fuse and Lock Bits may disable access to the device
Refer to device's datasheet about pinouts and programming details

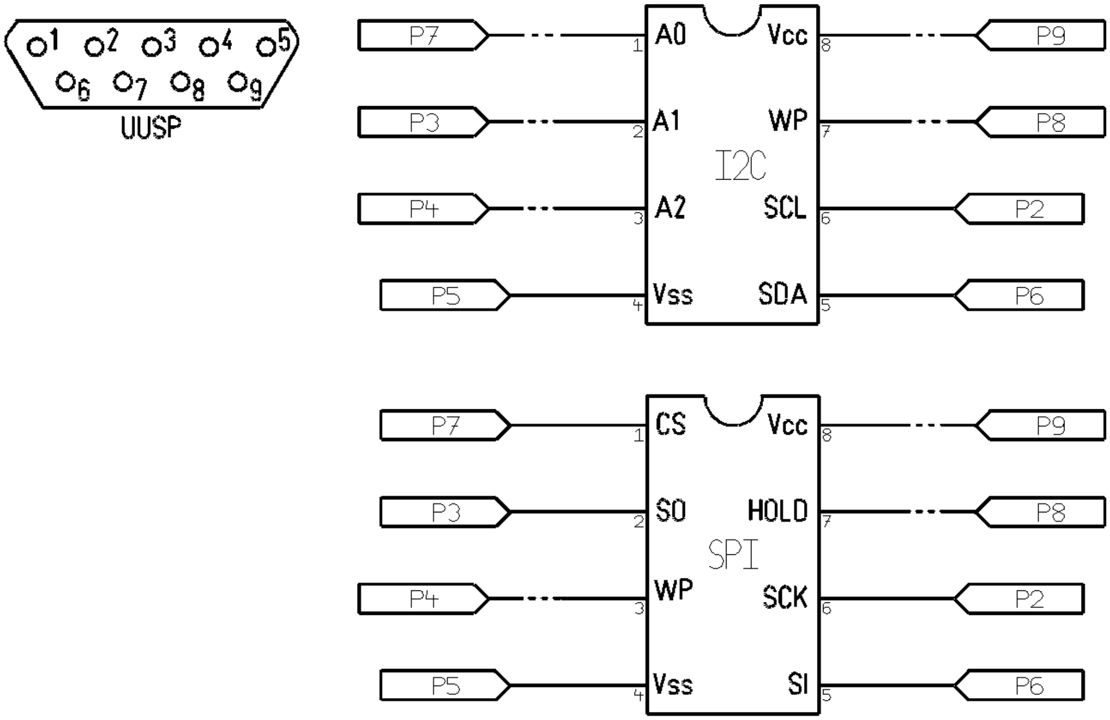
3.4 Microchip PICs



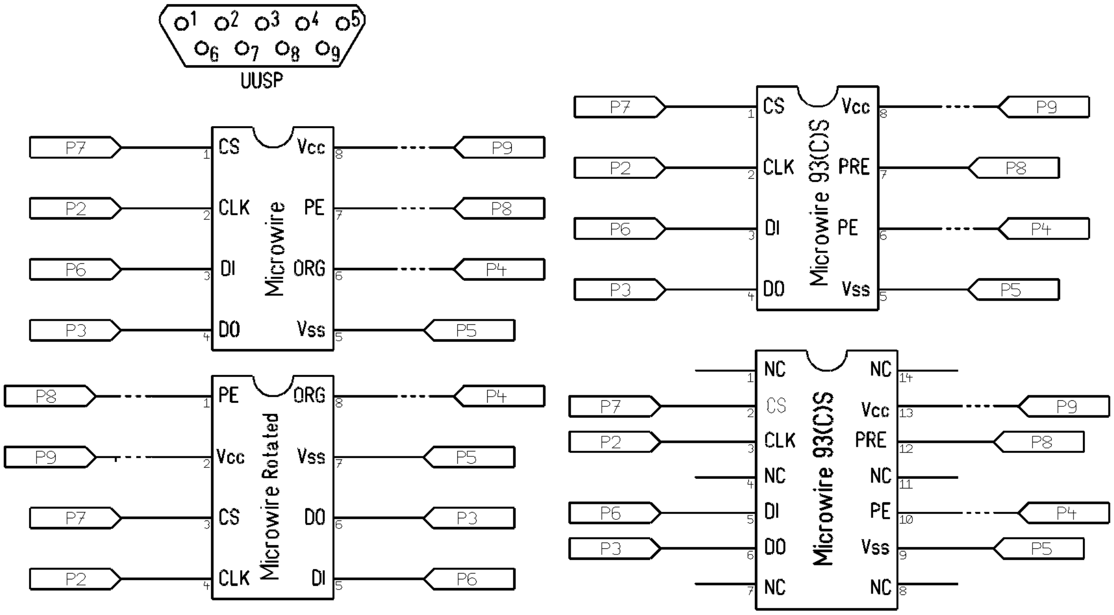
Refer to device's datasheet about pinouts and programming details

3.5 EEPROMs

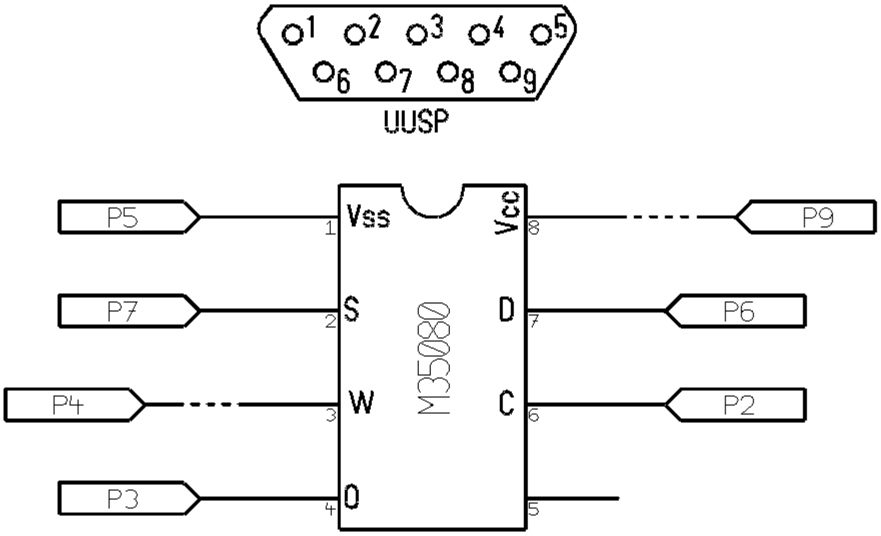
3.5.1 I2C and SPI



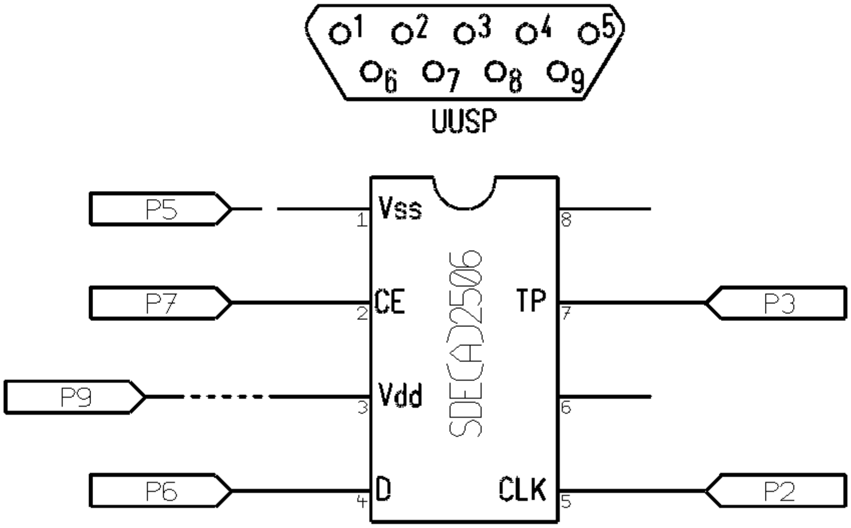
3.5.2 Microwire



3.5.3 M35080

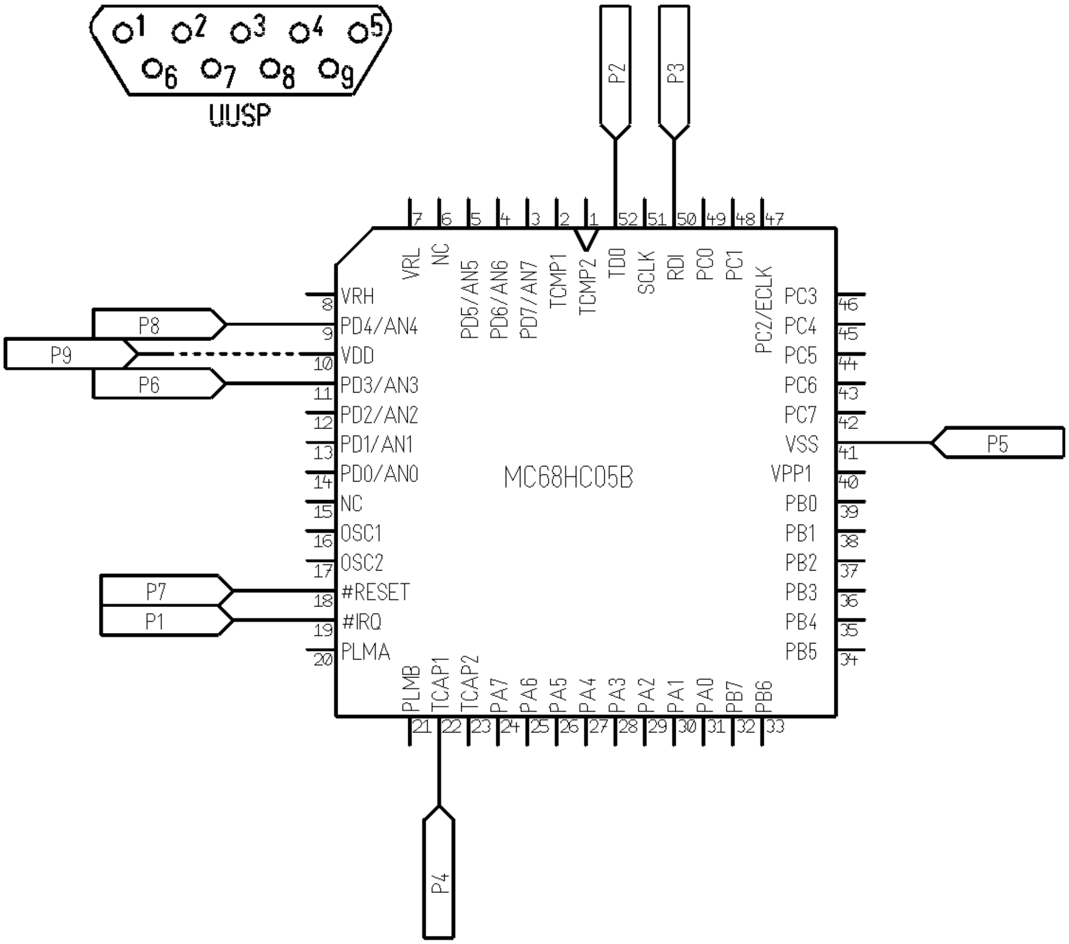


3.5.4 SDA(E)2506

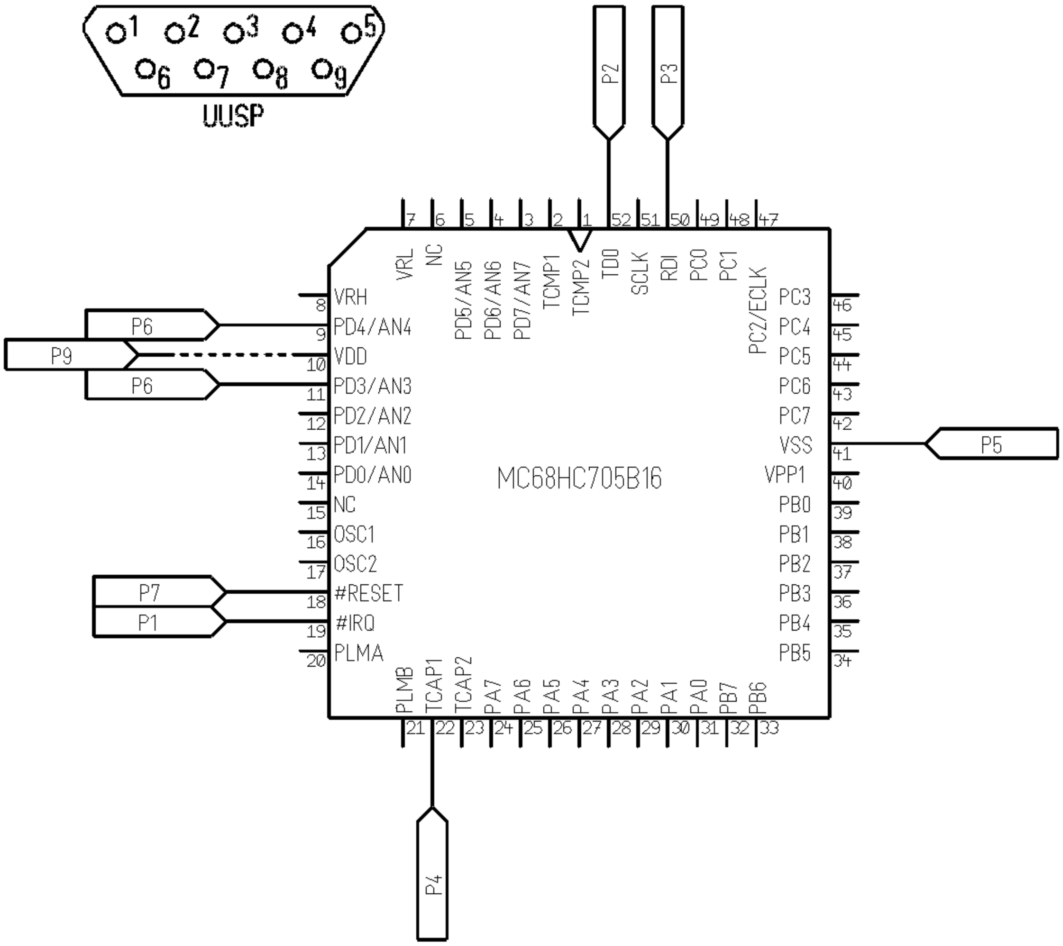


3.6 Motorola HC05

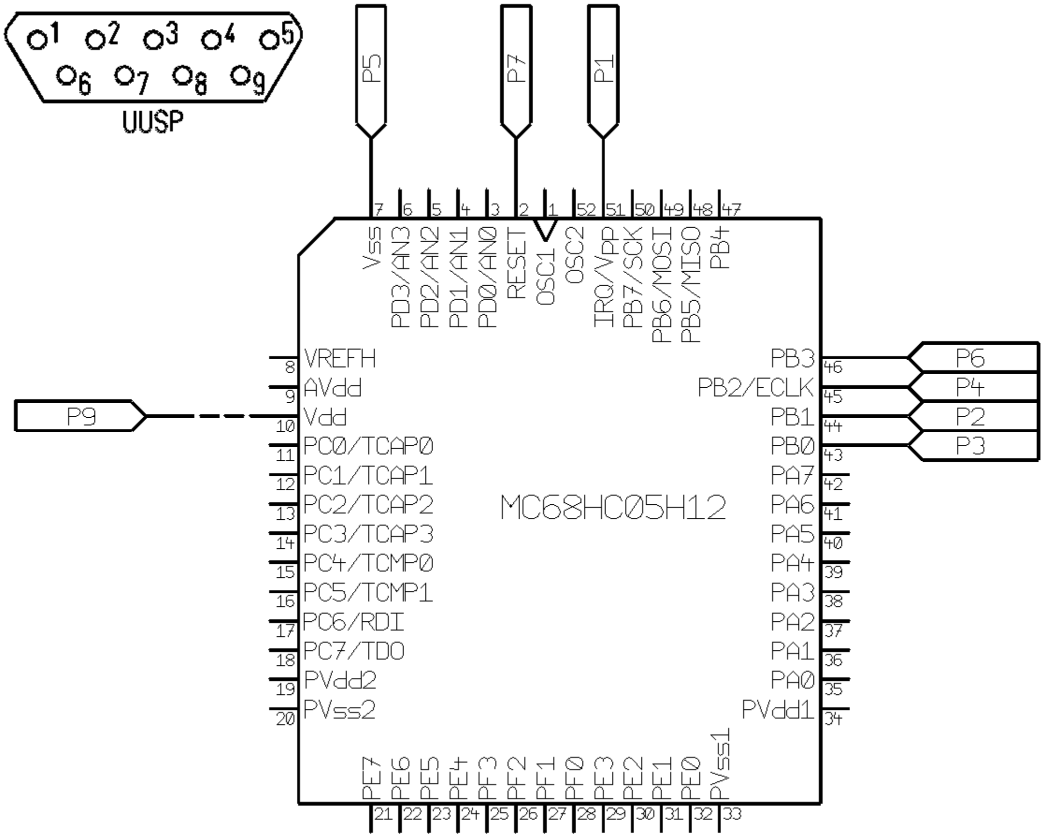
3.6.1 MC68HC05B PLCC52



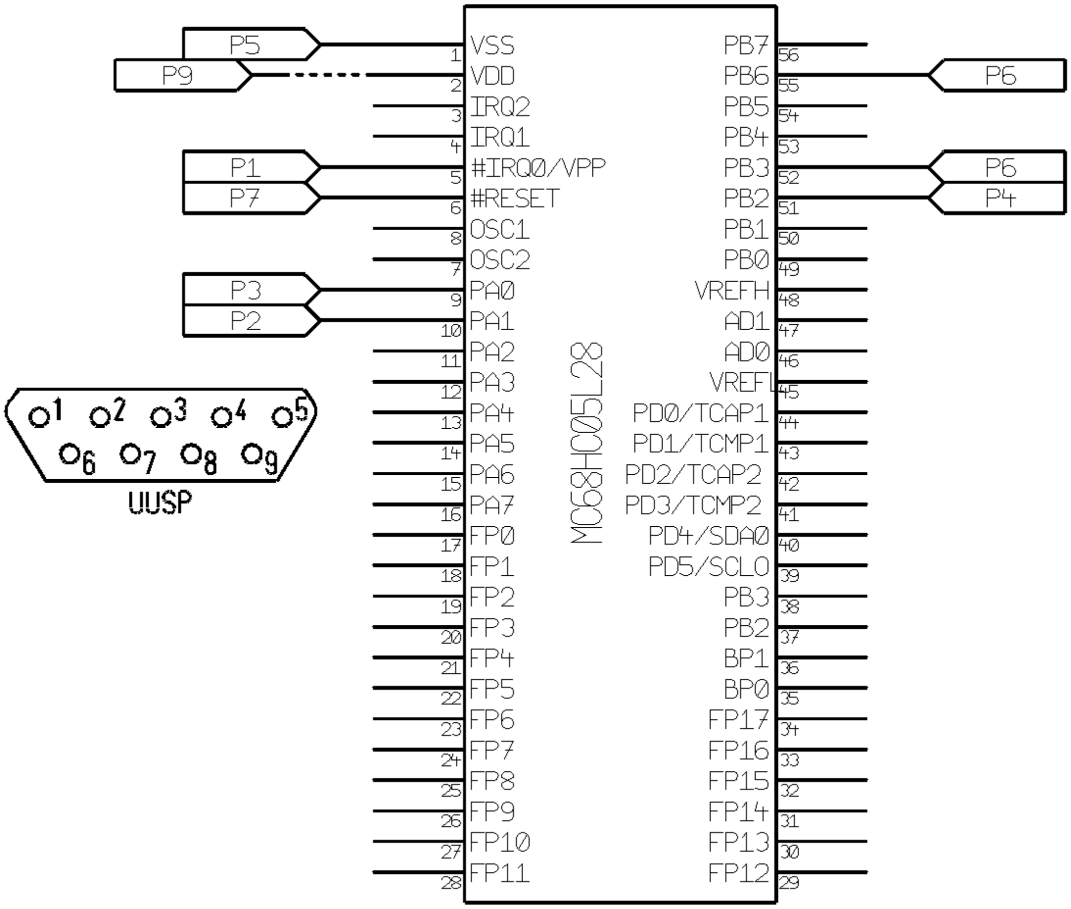
3.6.2 MC68HC705B16 PLCC52



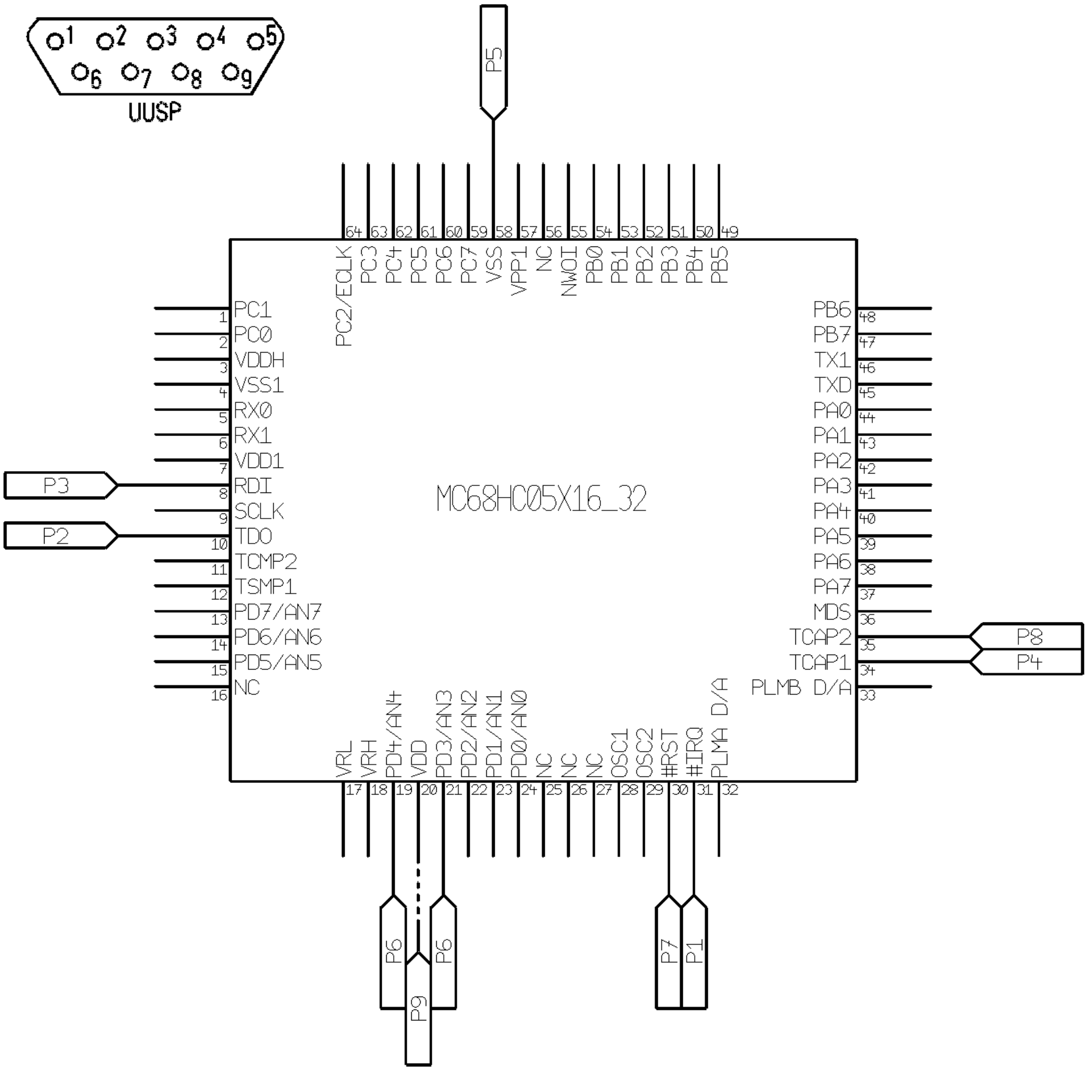
3.6.3 MC68HC05H12 PLCC52



3.6.4 MC68HC05L28 PDIP56

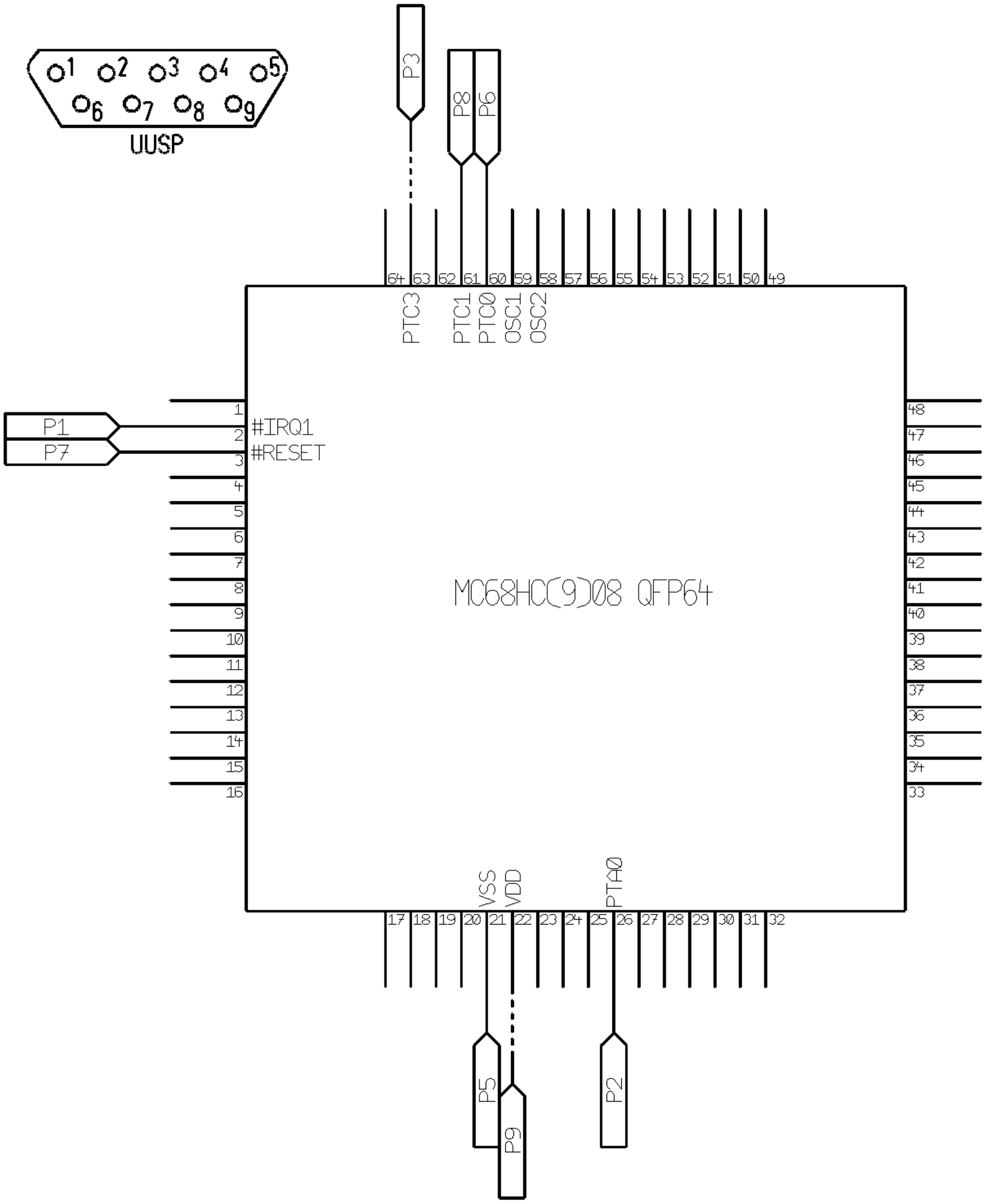


3.6.5 MC68HC05X16/32 QFP64

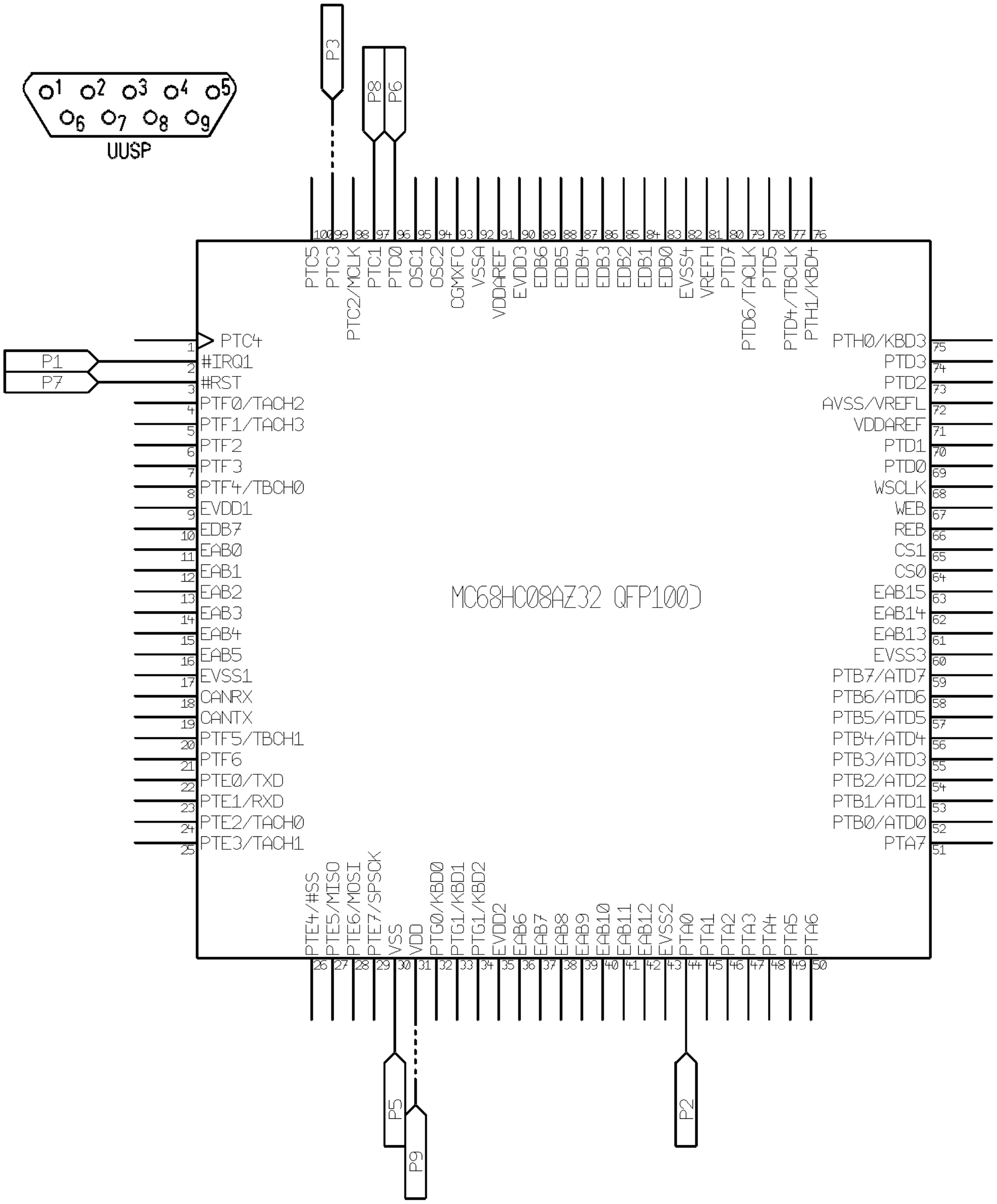


3.7 Motorola HC08

3.7.1 MC68HC(9)08 QFP64

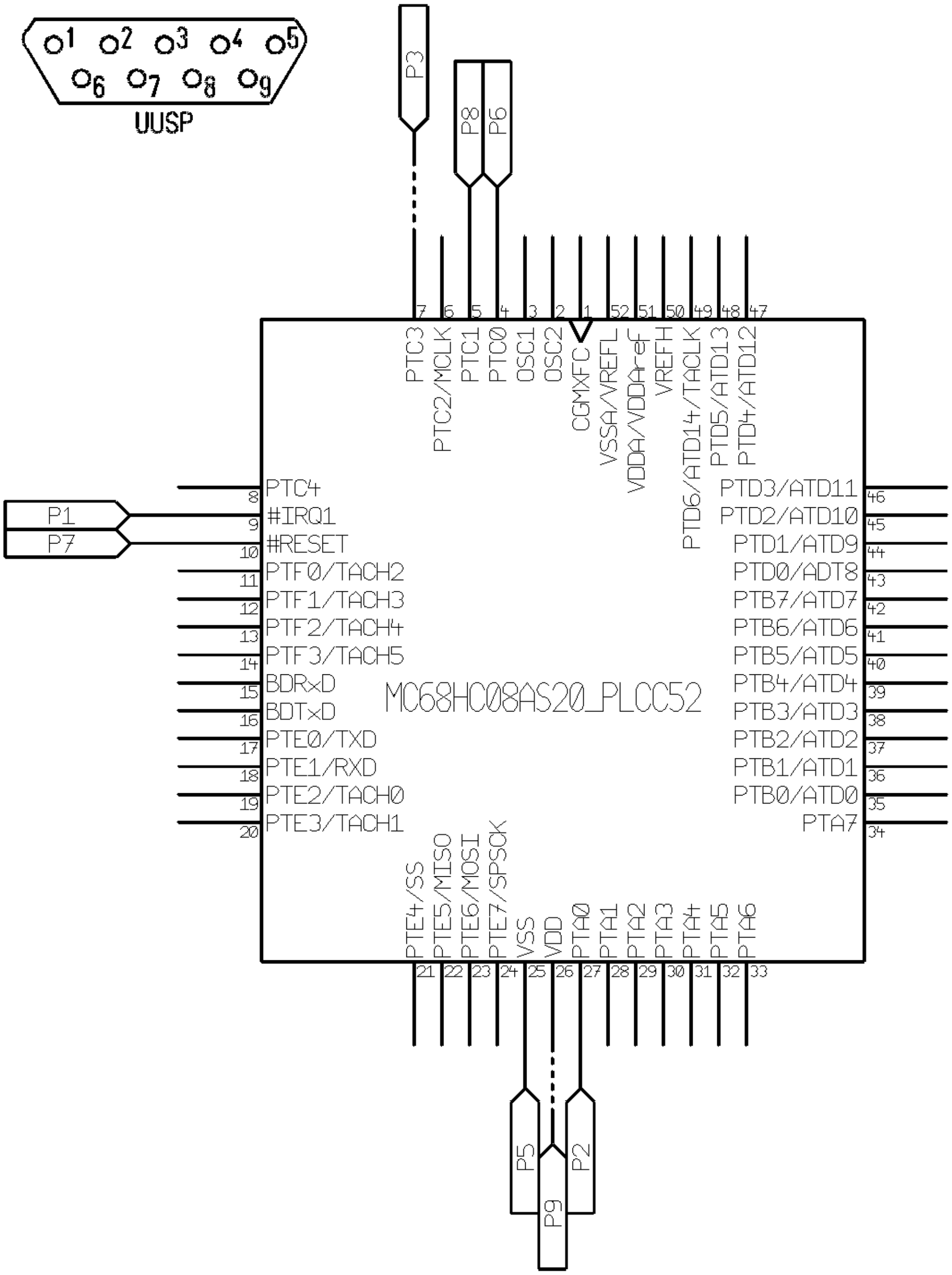


3.7.2 MC68HC08AZ32 QFP100



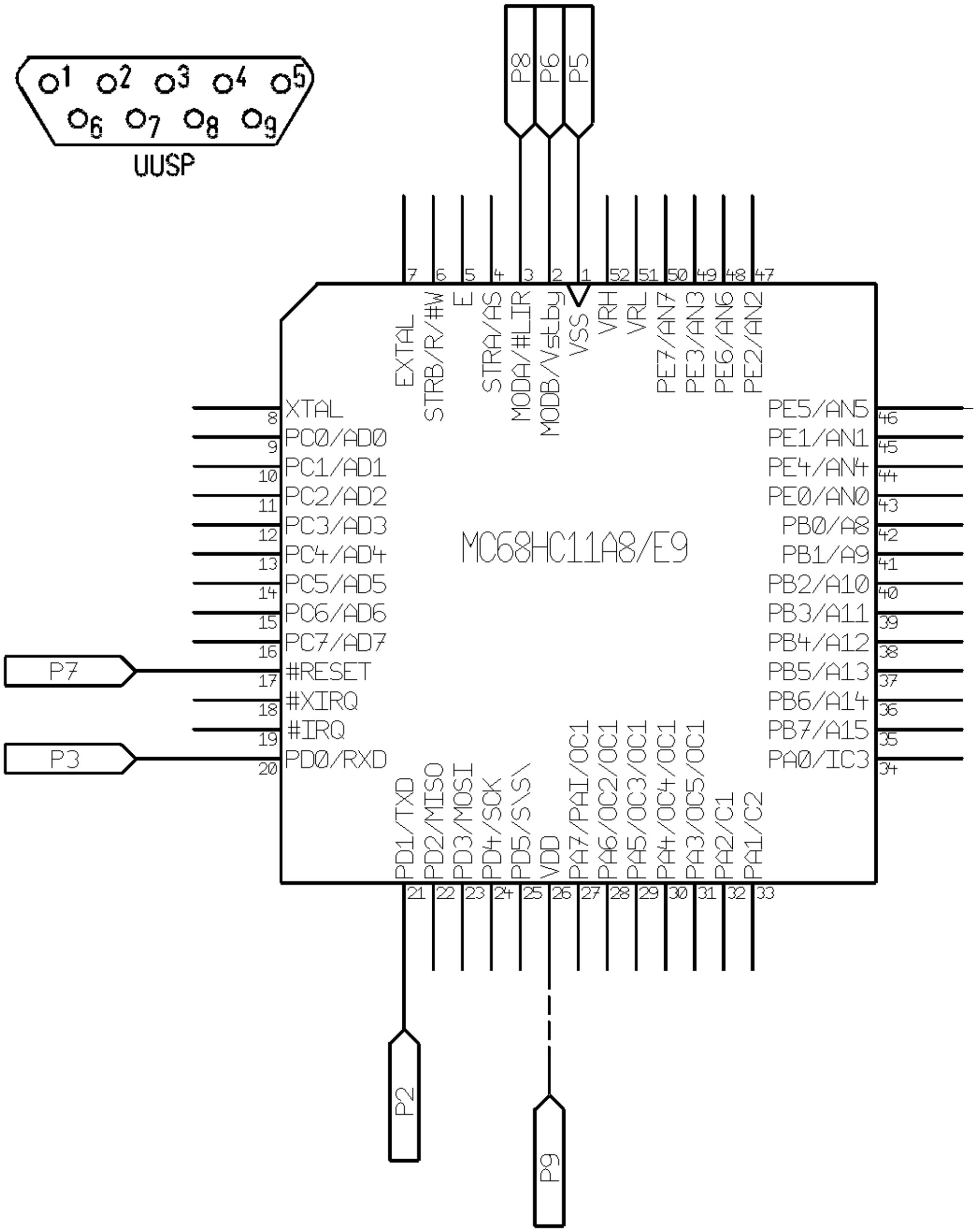
.

3.7.3 MC68HC08AS20 PLCC52

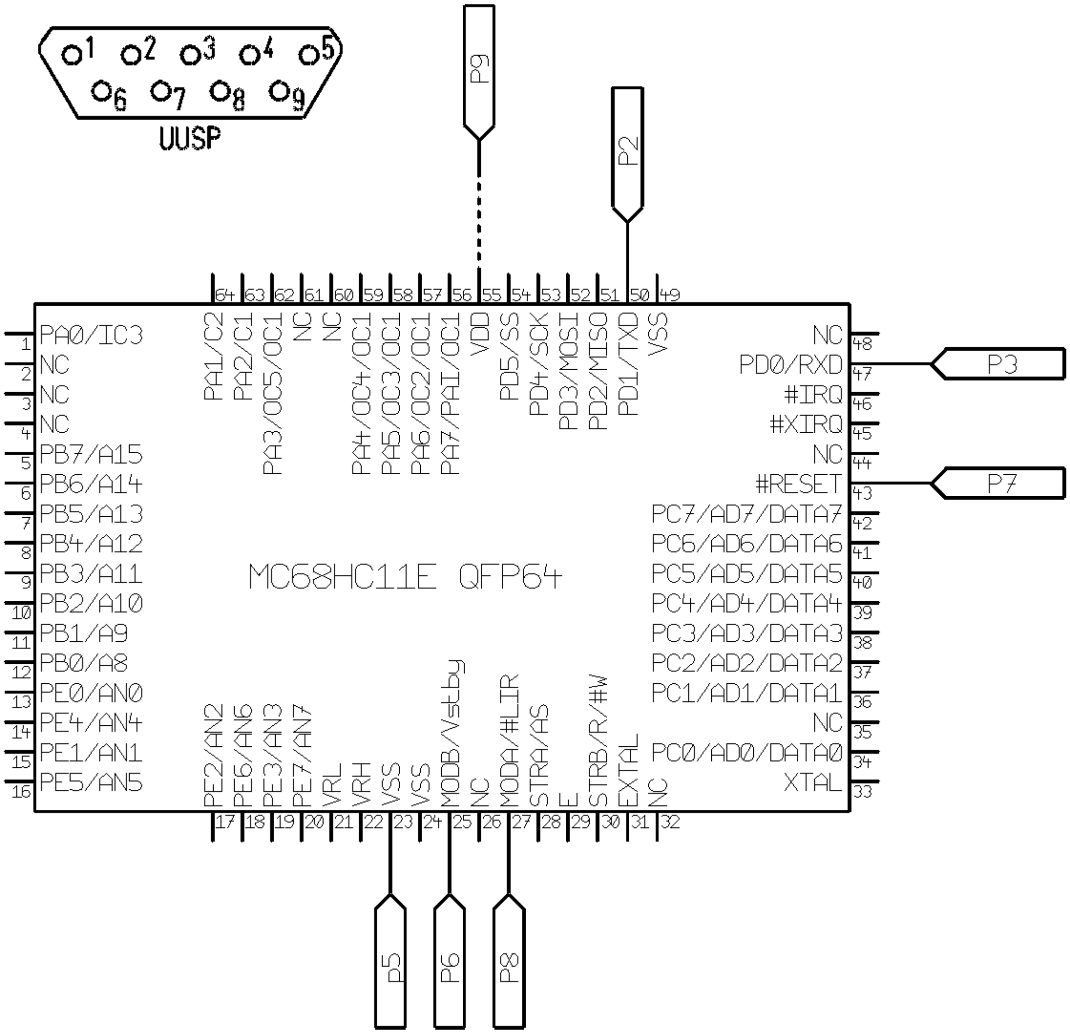


3.8 Motorola HC11

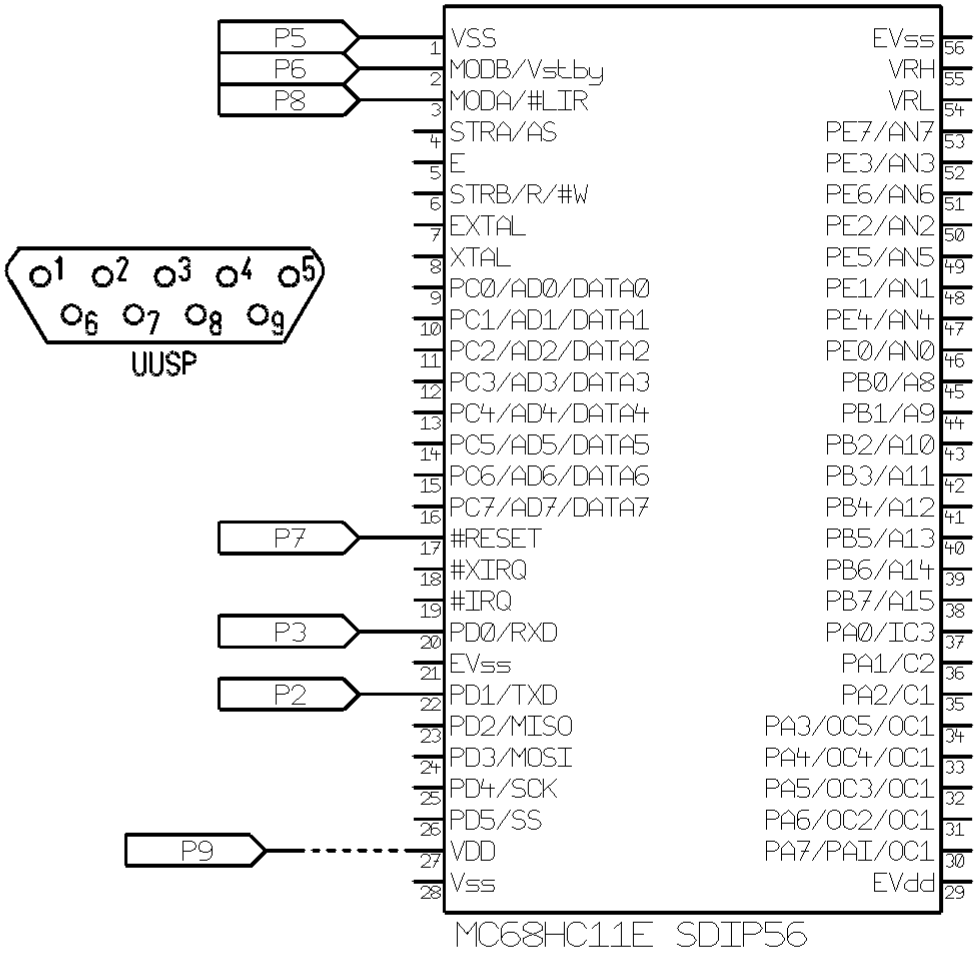
3.8.1 MC68HC11A8/E9 PLCC52



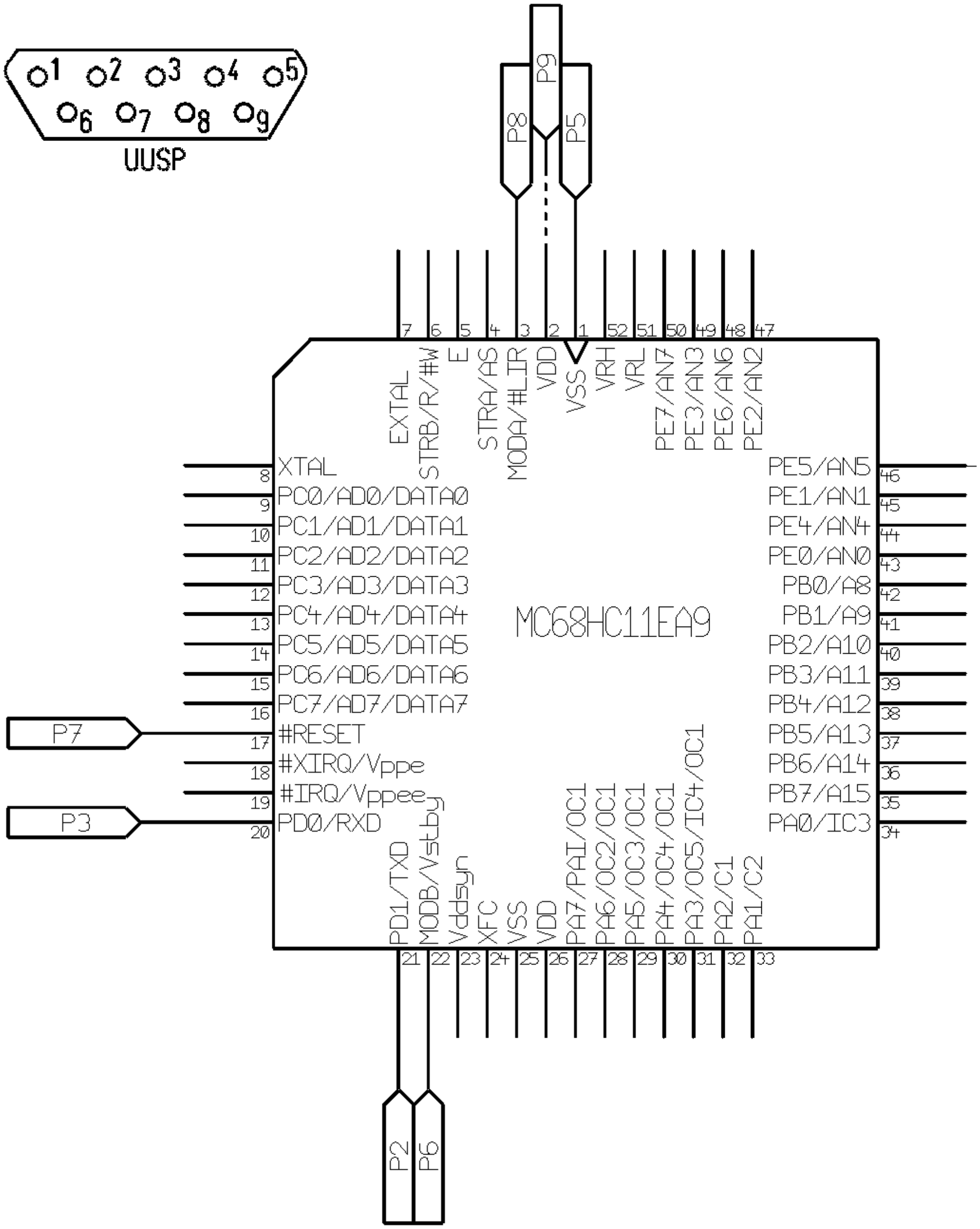
3.8.2 MC68HC11E QFP64



3.8.3 MC68HC11E SDIP56

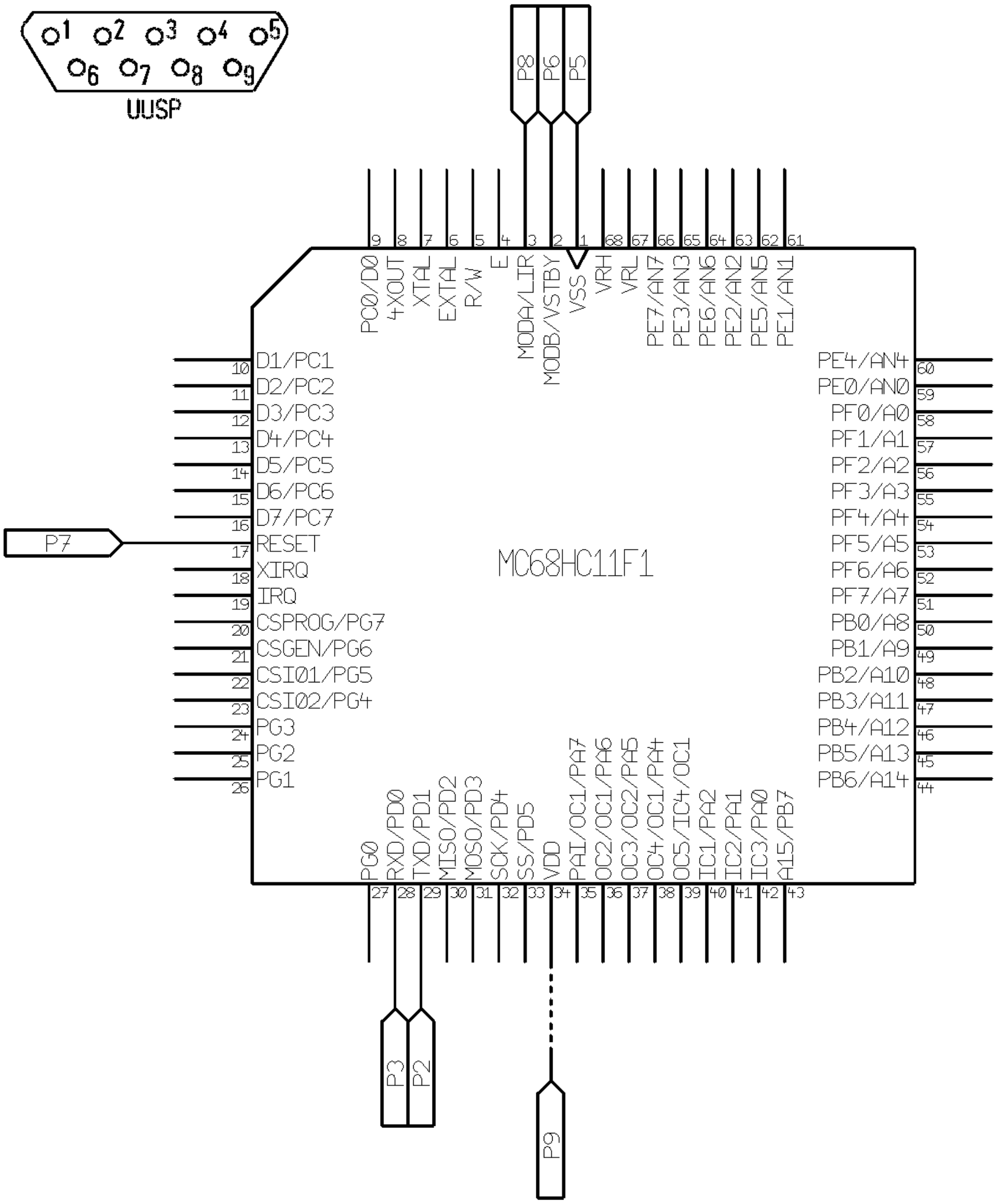


3.8.4 MC68HC11EA9 PLCC52

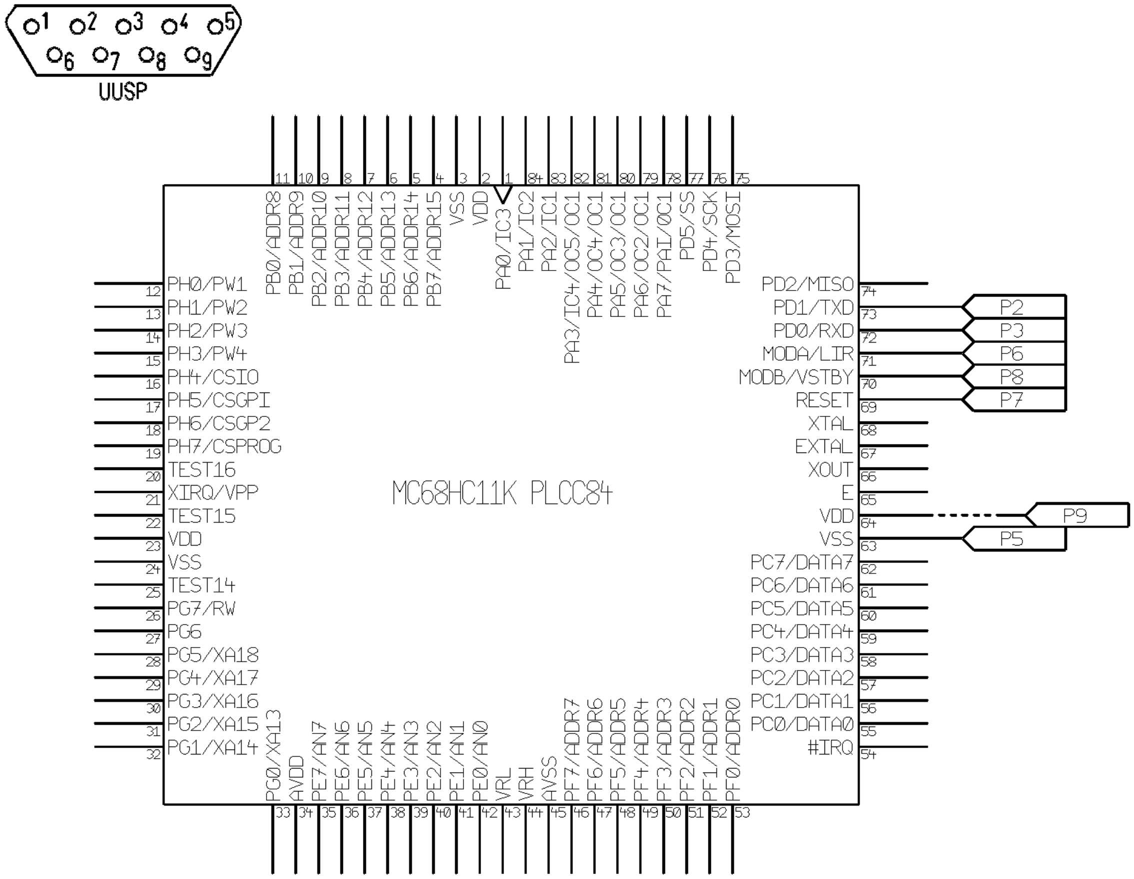


LIFT Vddsyn (23) pin
REPLACE ORIGINAL QUARTZ RESONATOR WITH A 8MHz ONE
See Application Note: EB422.PDF available from www.freescale.com

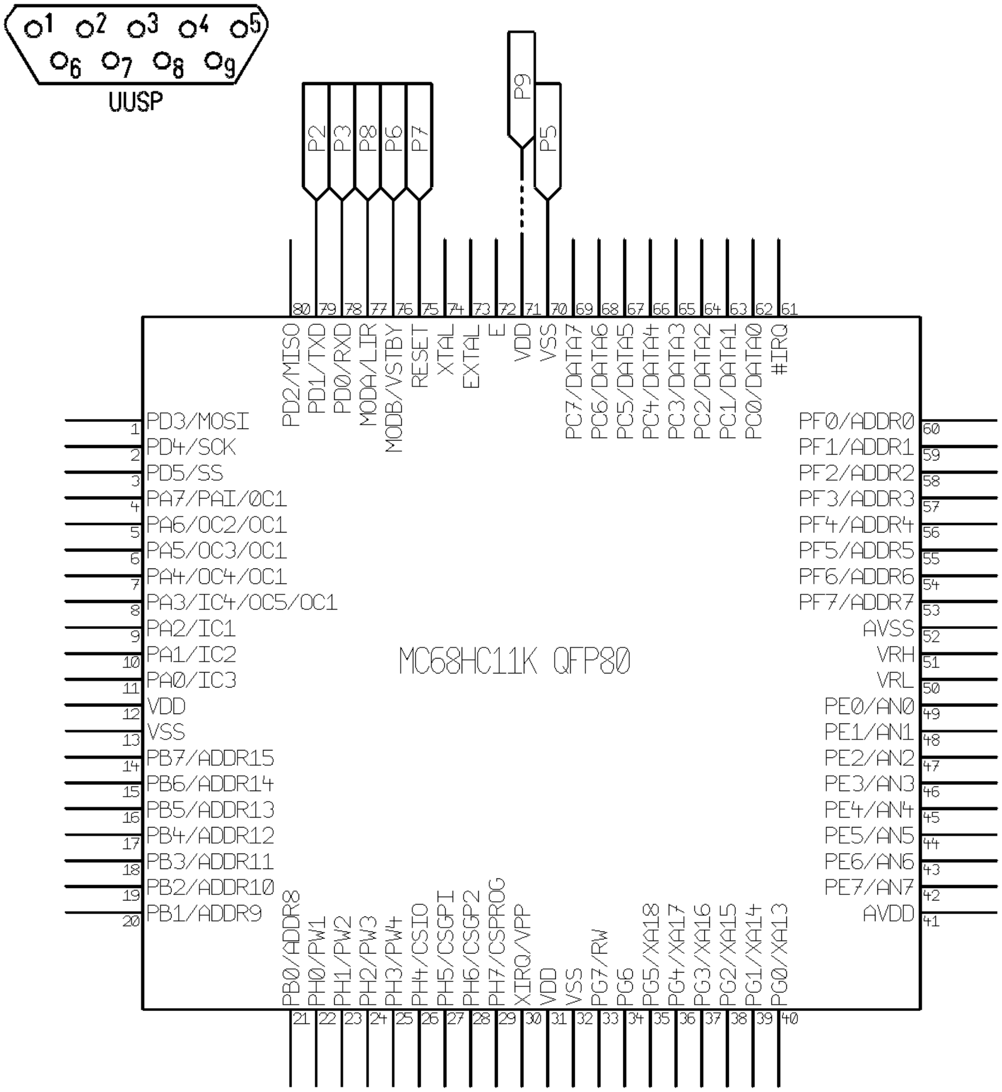
3.8.5 MC68HC11F1 PLCC68



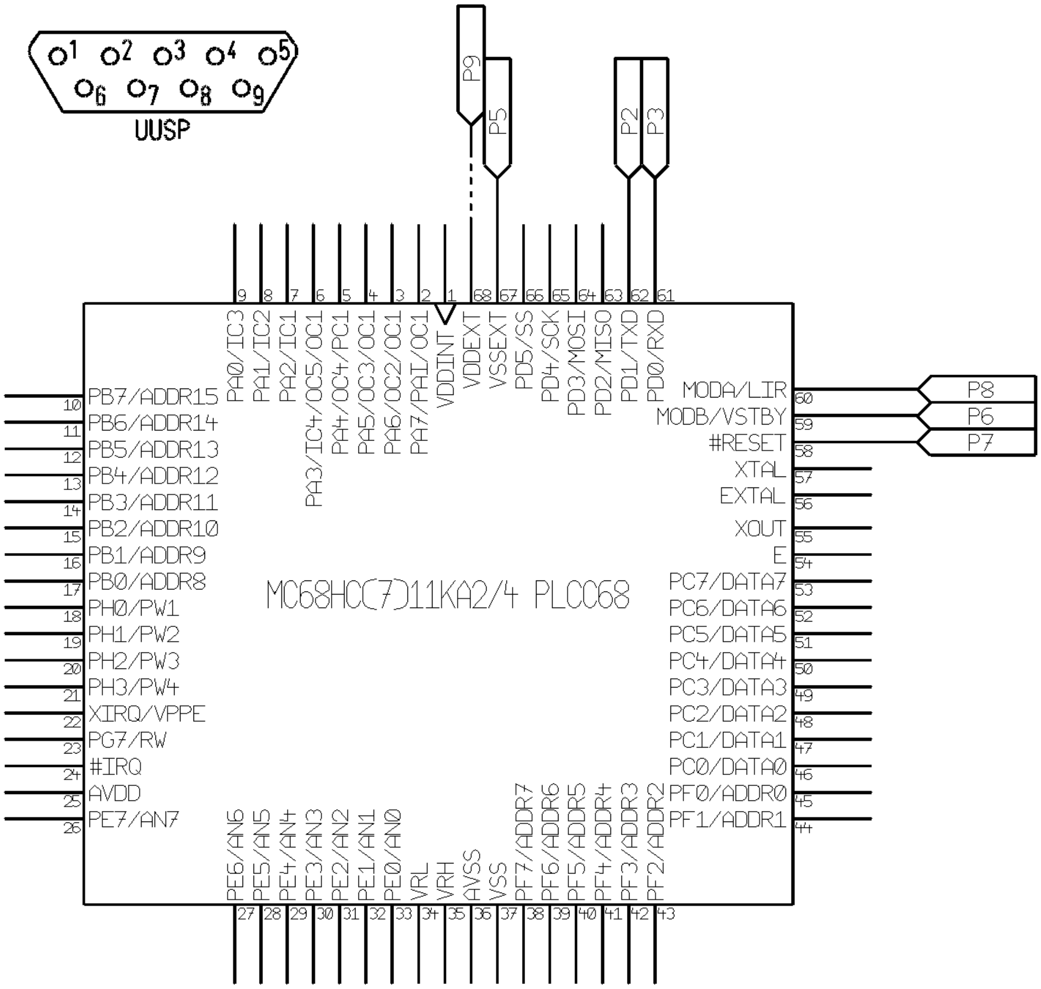
3.8.6 MC68HC11K PLCC84



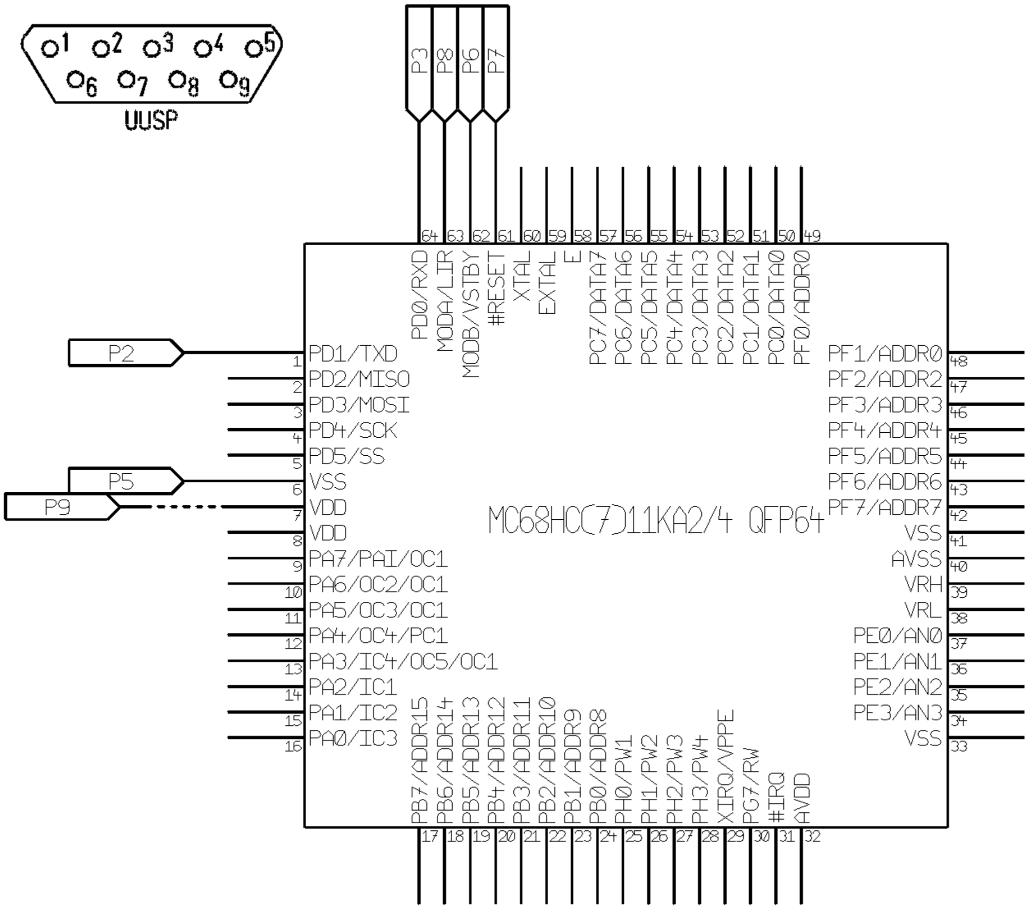
3.8.7 MC68HC11K QFP80



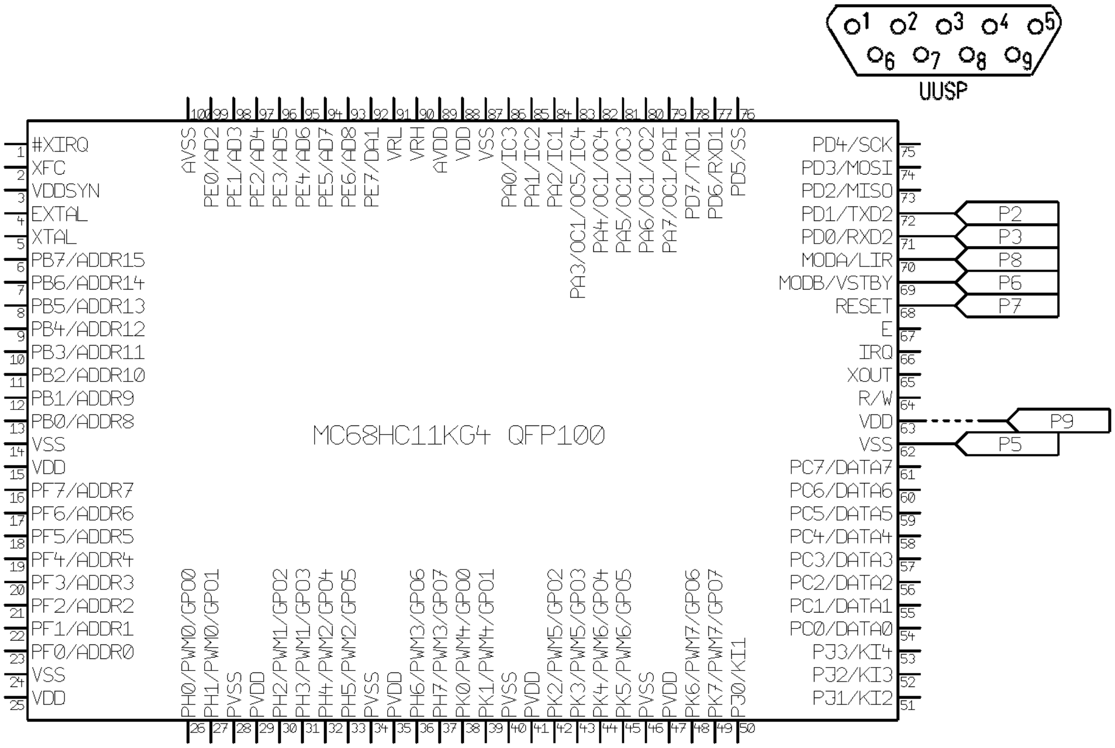
3.8.8 MC68HC11KA2/4 PLCC68



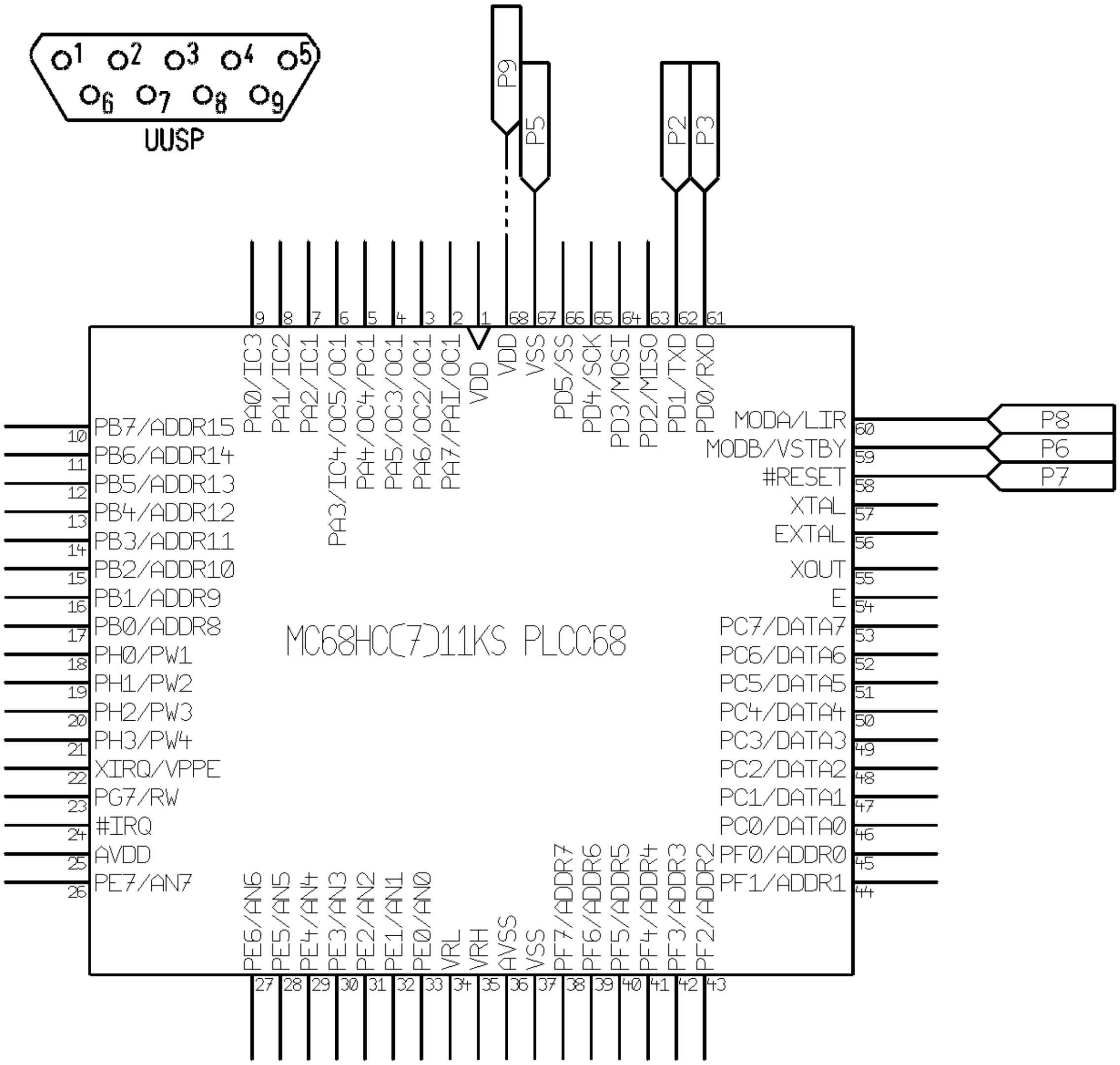
3.8.9 MC68HC11KA2/4 QFP64



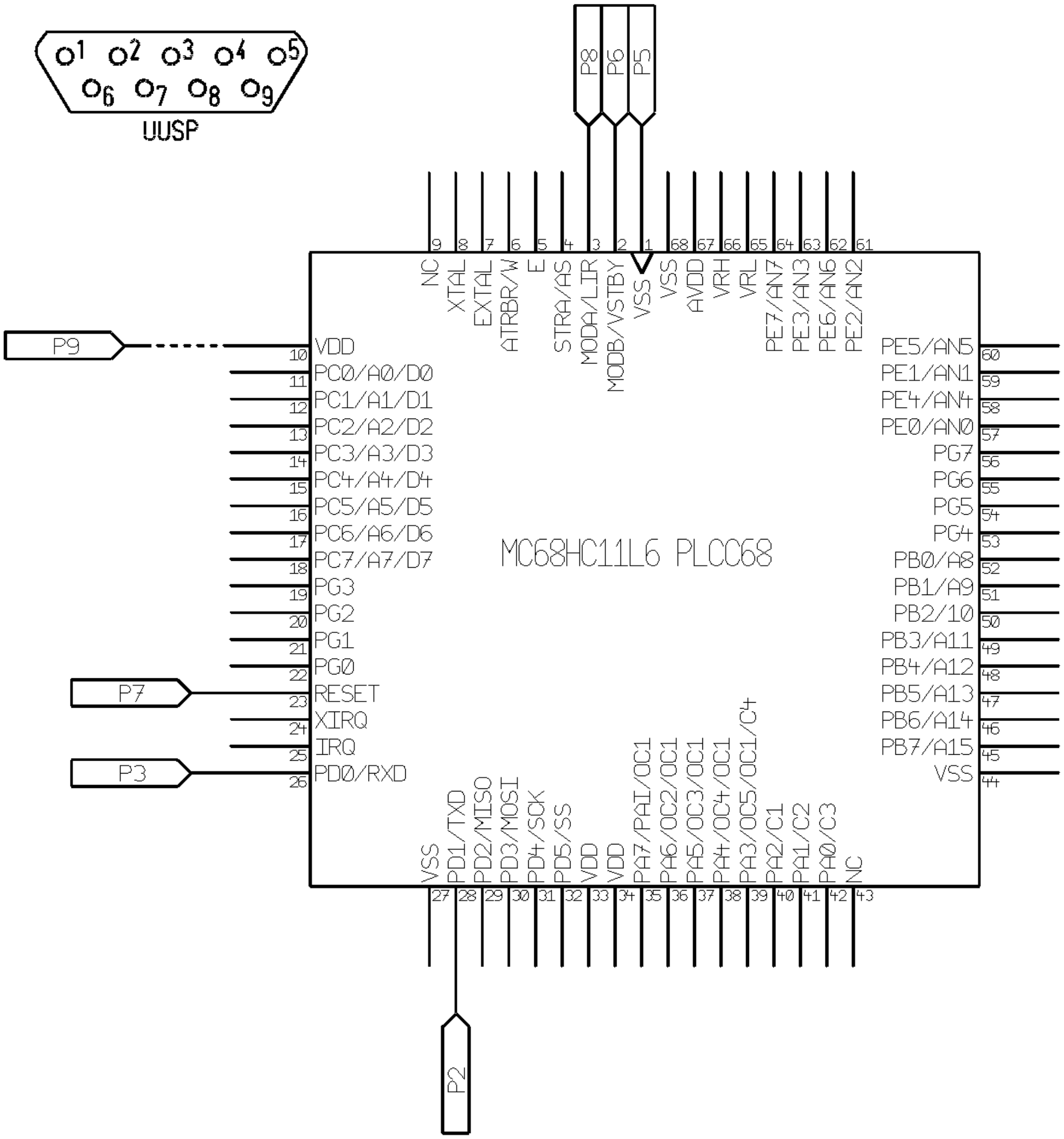
3.8.10 MC68HC11KG4 QFP100



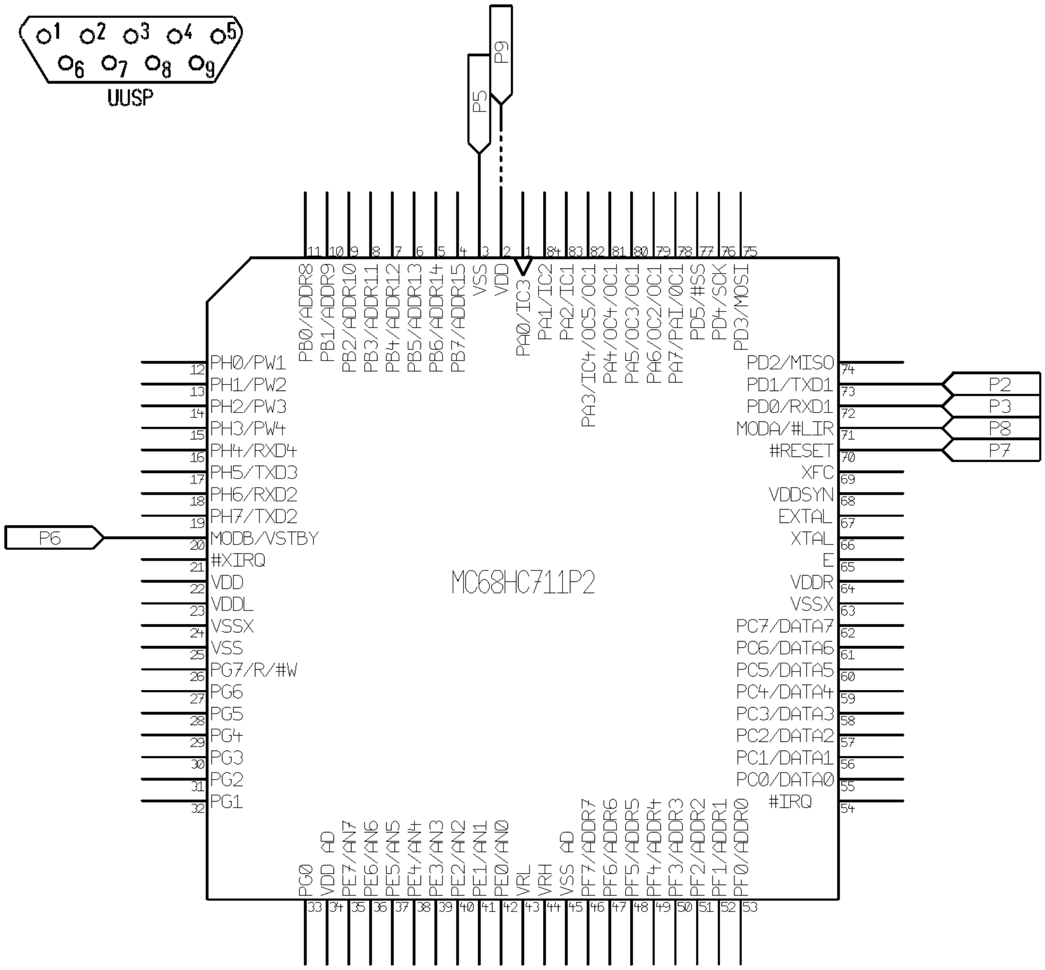
3.8.11 MC68HC11KS PLCC68



3.8.12 MC68HC11L6 PLCC68

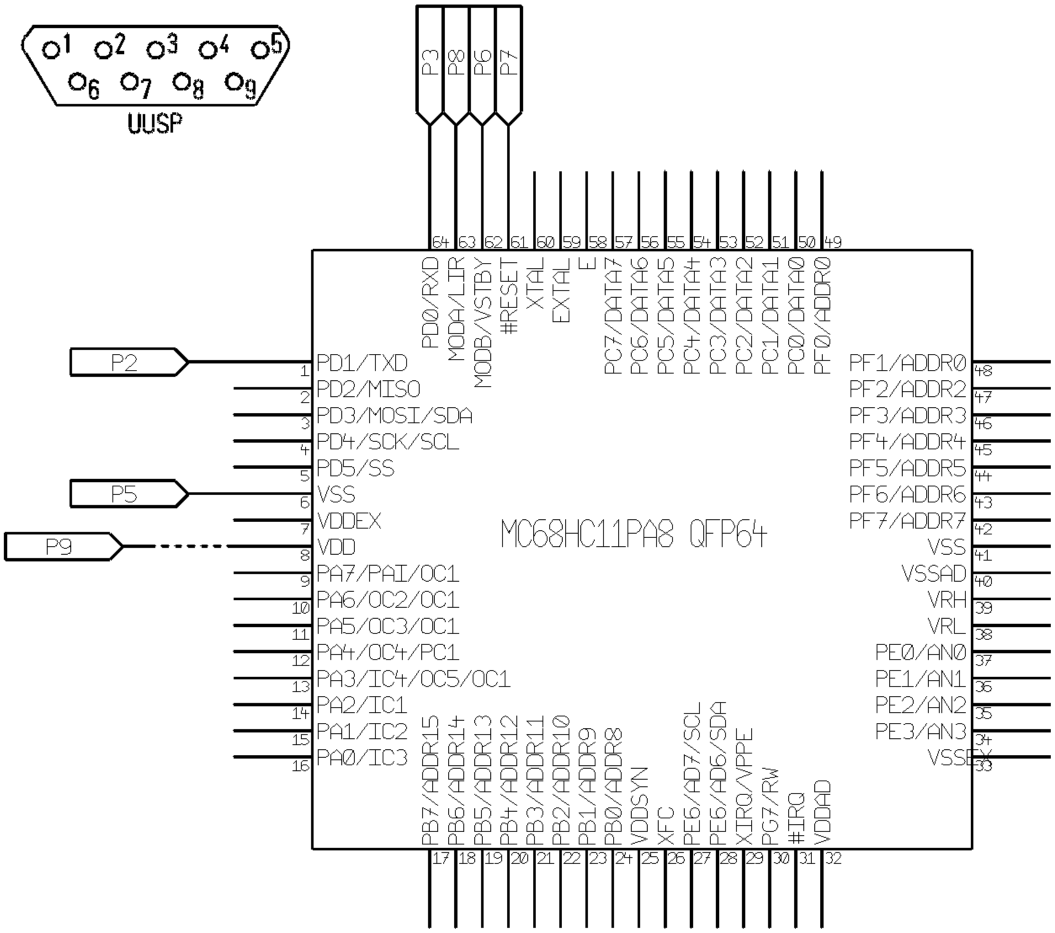


3.8.13 MC68HC11P2 PLCC84

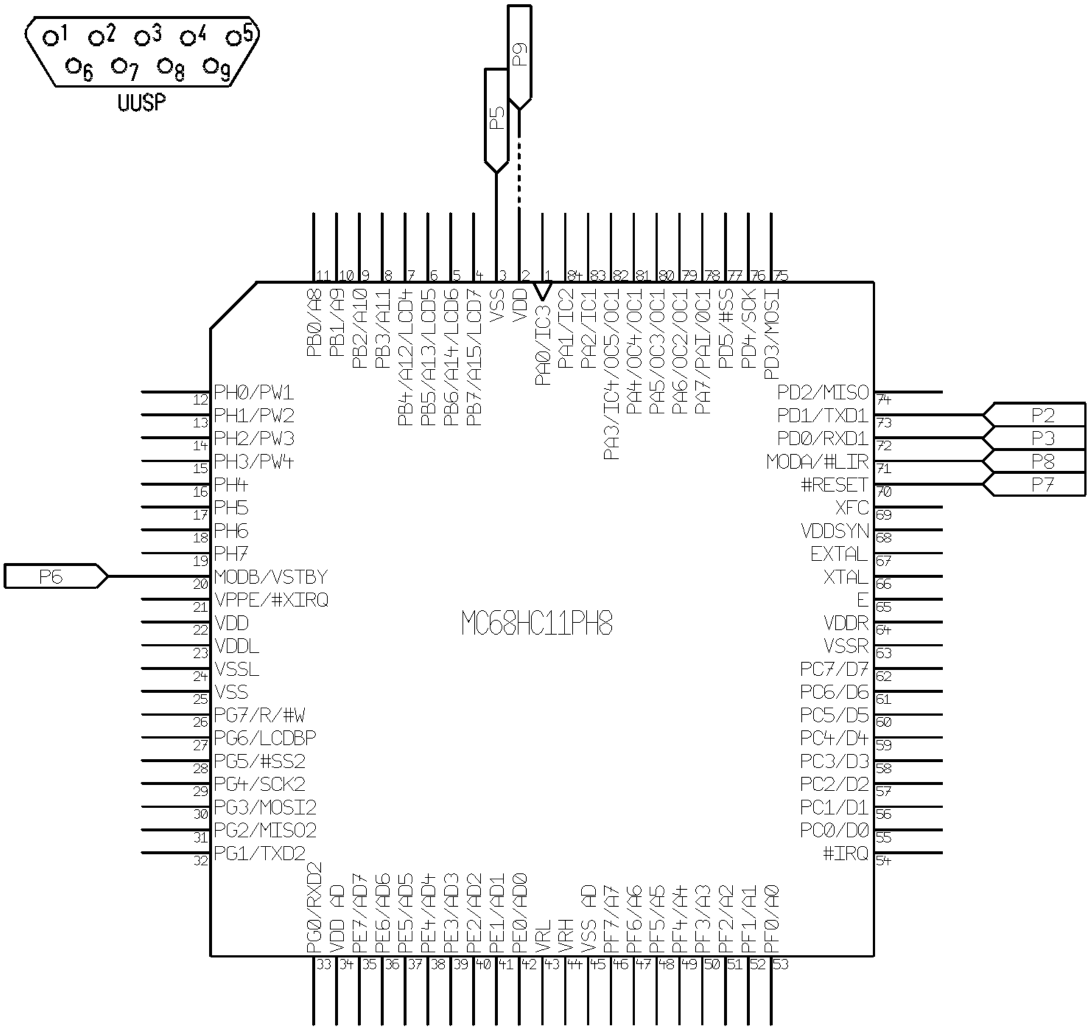


LIFT Vddsyn (68) pin
REPLACE ORIGINAL QUARTZ RESONATOR WITH A 8MHz ONE
See Application Note: EB422.PDF available from www.freescale.com

3.8.14 MC68HC11PA8 QFP64



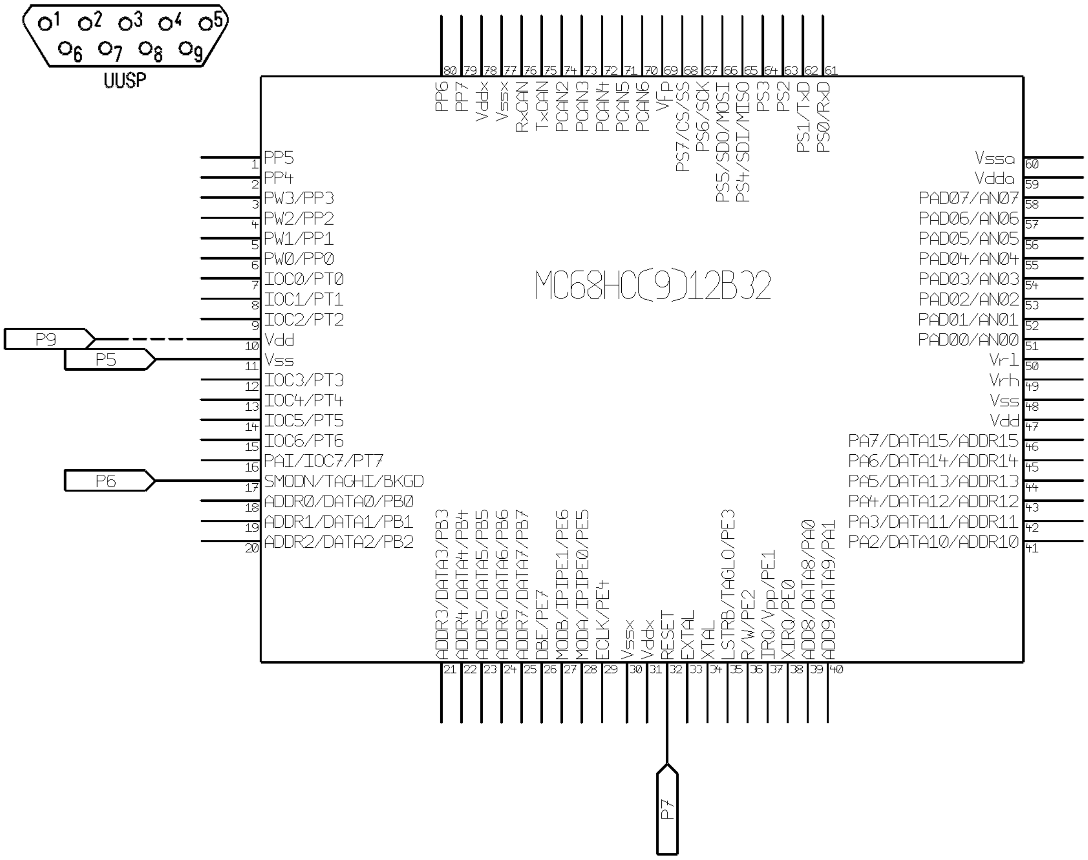
3.8.15 MC68HC11PH8 PLCC84



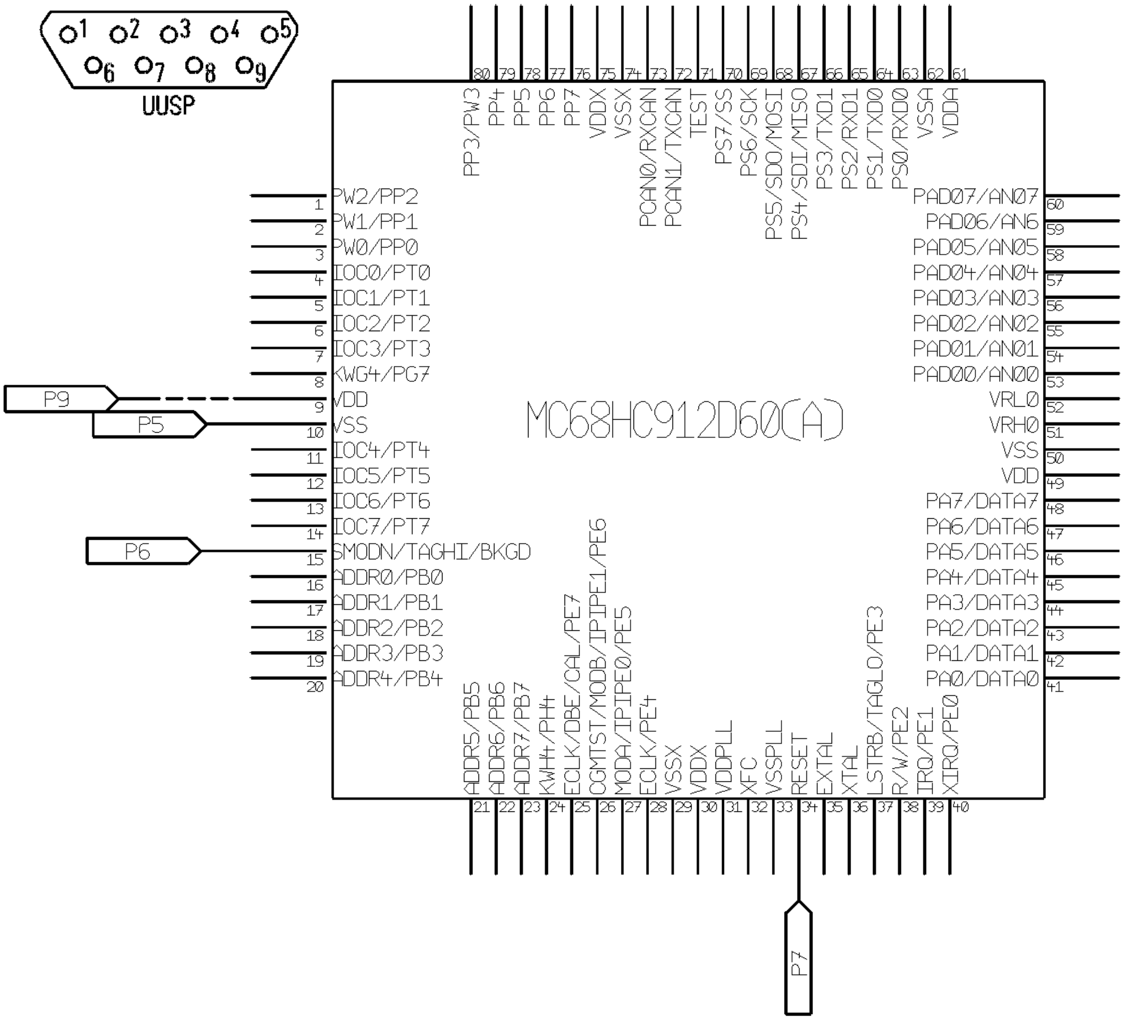
LIFT Vddsyn (68) pin
REPLACE ORIGINAL QUARTZ RESONATOR WITH A 8MHz ONE
See Application Note: EB422.PDF available from www.freescale.com

3.9 Motorola HC12

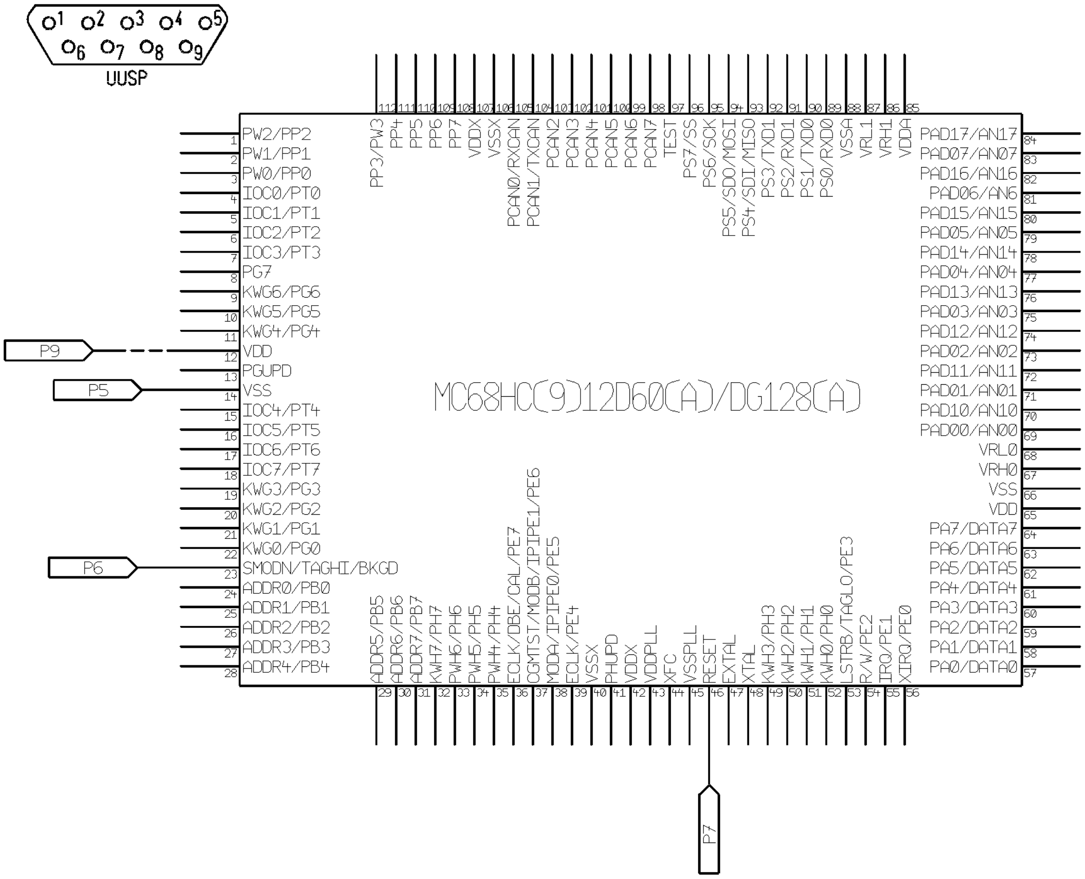
3.9.1 MC68HC(9)12B32 QFP80



3.9.2 MC68HC(9)12D60(A) QFP80

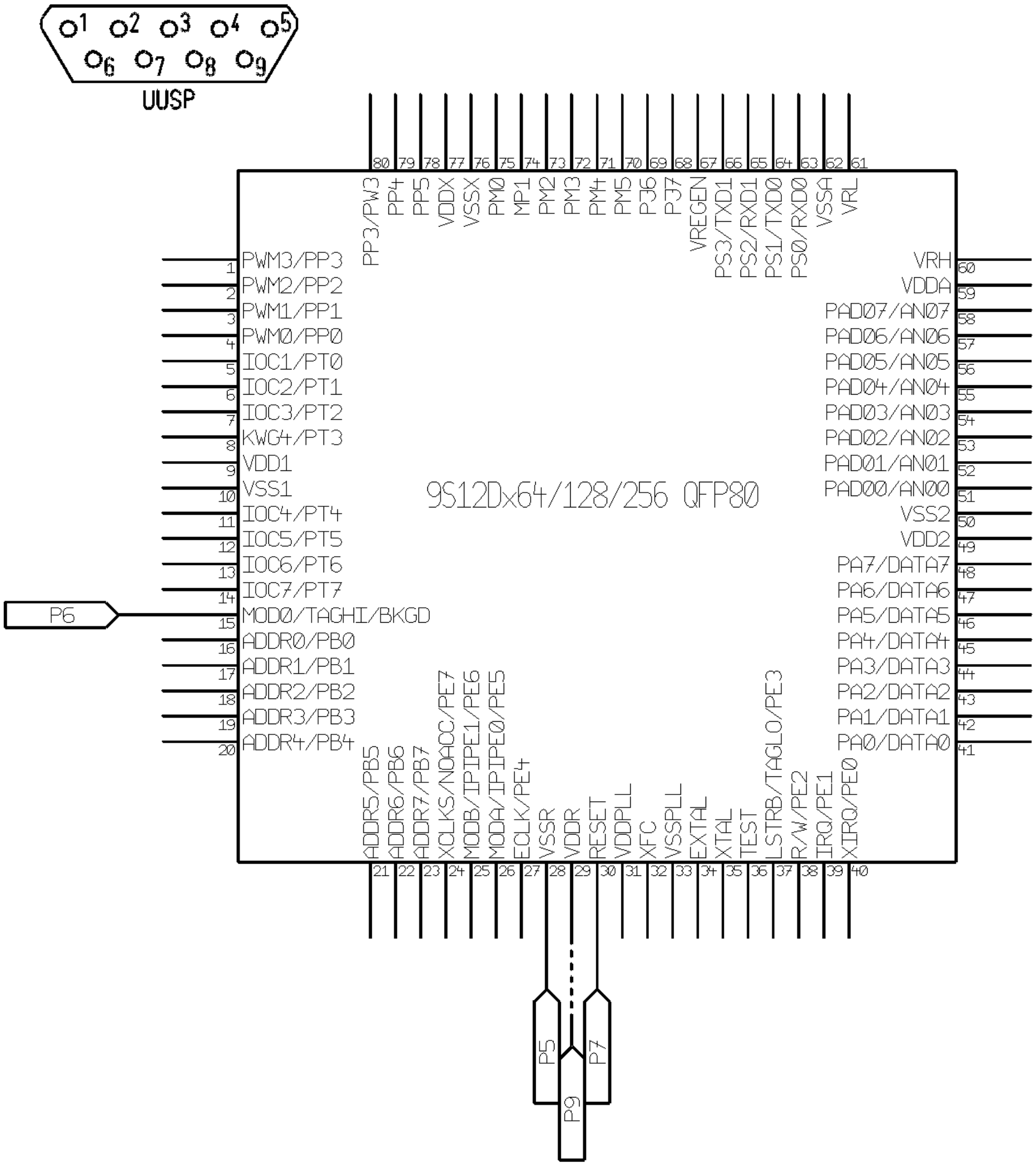


3.9.3 MC68HC(9)12D60(A)/DG128(A) QFP112

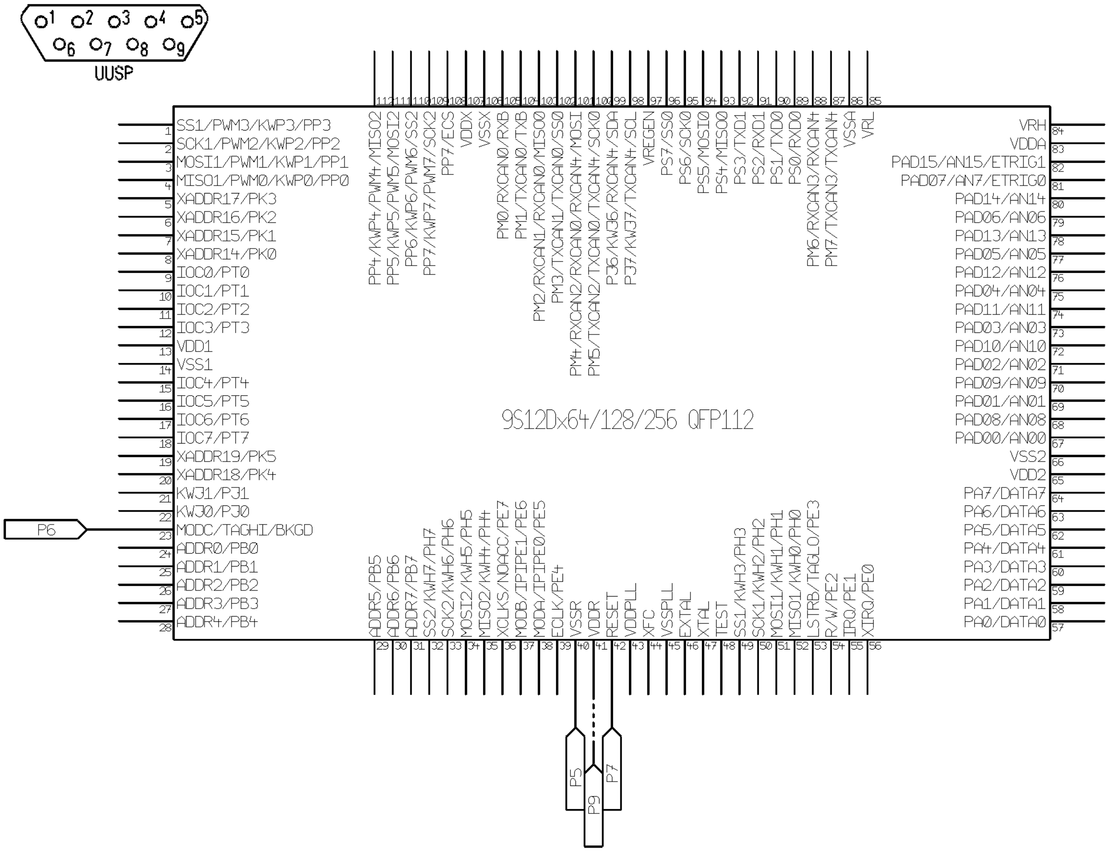


3.10 Motorola HCS12

3.10.1 MC9S12Dx64/128/256 QFP80



3.10.2 MC9S12Dx64/128/256 QFP112



3.11 78K0/HC912 Adapter

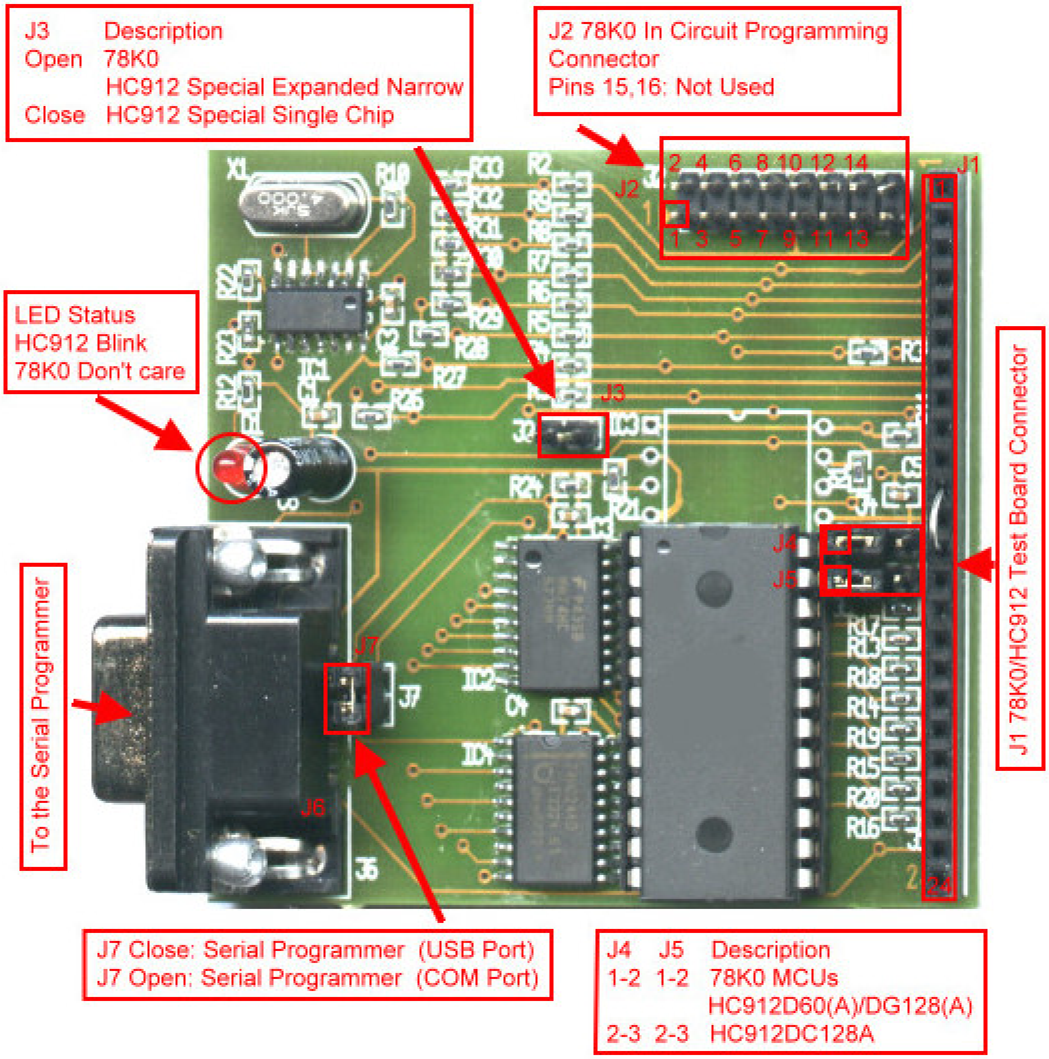
Supported Devices
NEC uPD780828A, uPD780973/4, uPD780948/9
Motorola (Freescale) 68HC912D60(A)DG128(A), 68HC912DC128A

78K0 in circuit programming via 14 wire connection using J2 connector (Pins 15 and 16 are not used)
uPD780828A connections
uPD780973/4 connections
uPD780948/9 connections

78K0 programming by a test board
solder the MCU on a test board, and plug it into the 24 pin connector J1

HC912 programming by a test board
solder the MCU on a test board, and plug it into the 24 pin connector J1
LED must blink during read/program

3.11.1 Jumpers and Connectors Description



J3	Description
Open	78K0 HC912 Special Expanded Narrow
Close	HC912 Special Single Chip

J2 78K0 In Circuit Programming Connector
Pins 15,16: Not Used

LED Status
HC912 Blink
78K0 Don't care

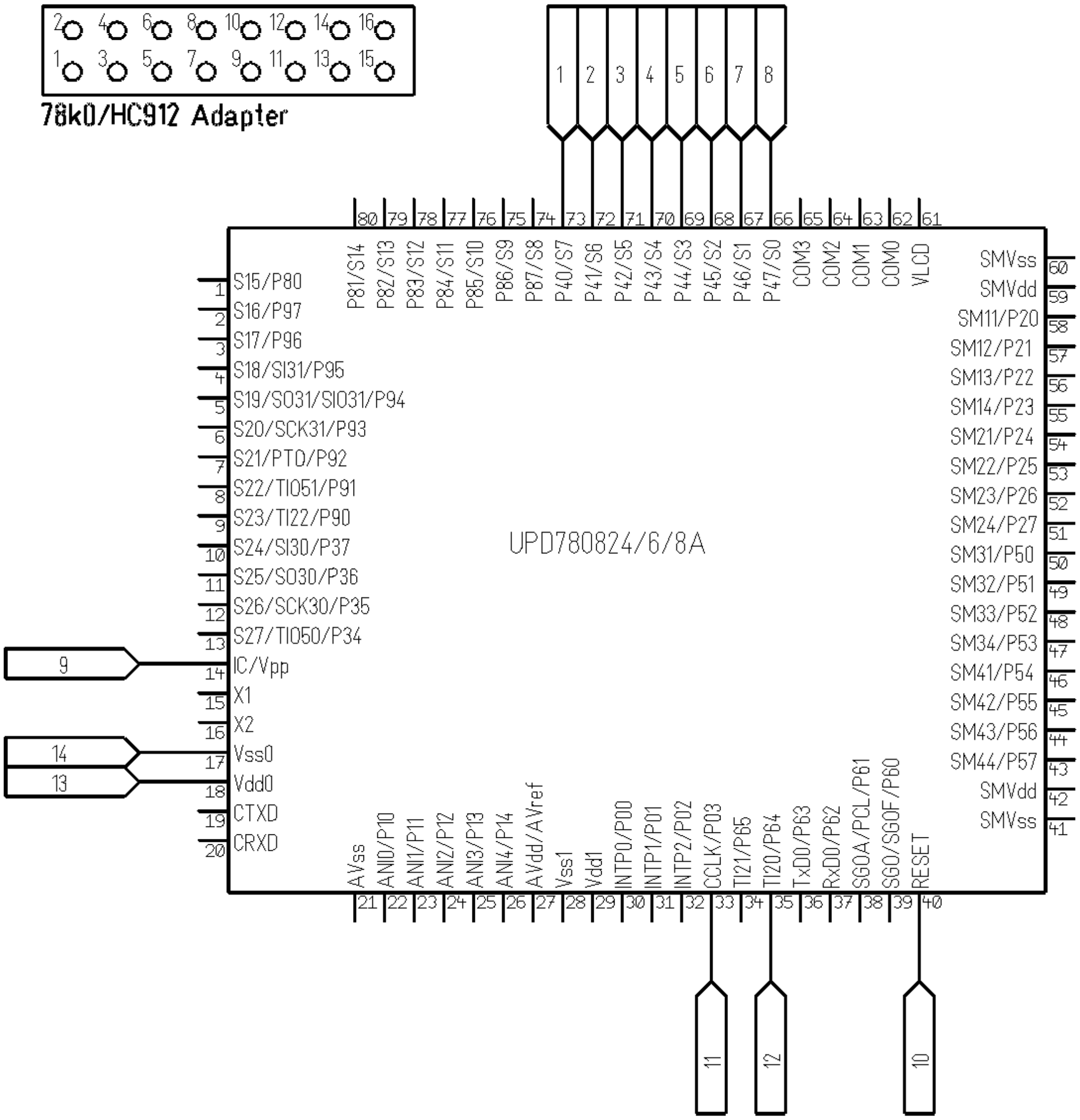
To the Serial Programmer

J1 78K0/HC912 Test Board Connector

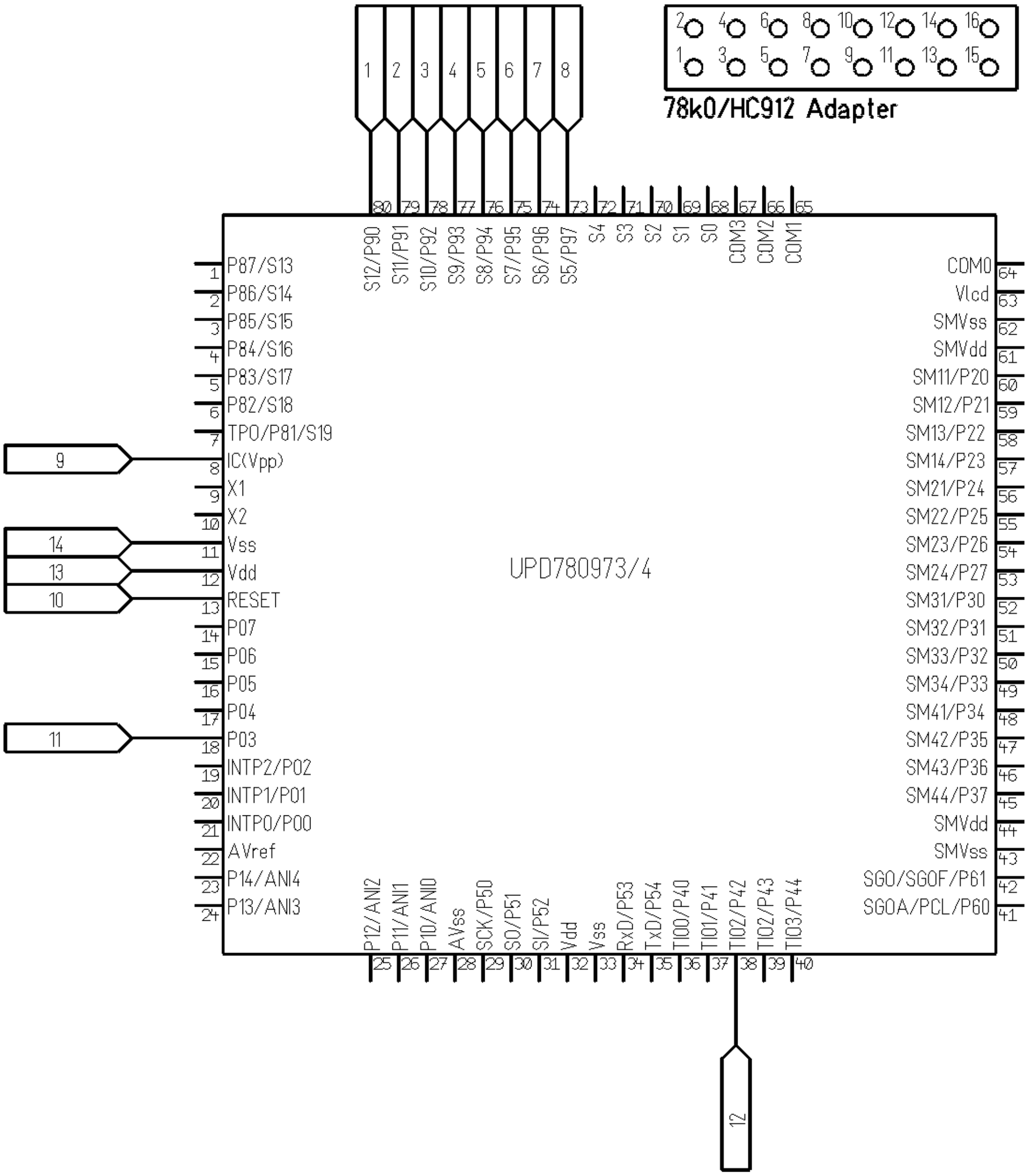
J7 Close: Serial Programmer (USB Port)
J7 Open: Serial Programmer (COM Port)

J4	J5	Description
1-2	1-2	78K0 MCUs HC912D60(A)/DG128(A)
2-3	2-3	HC912DC128A

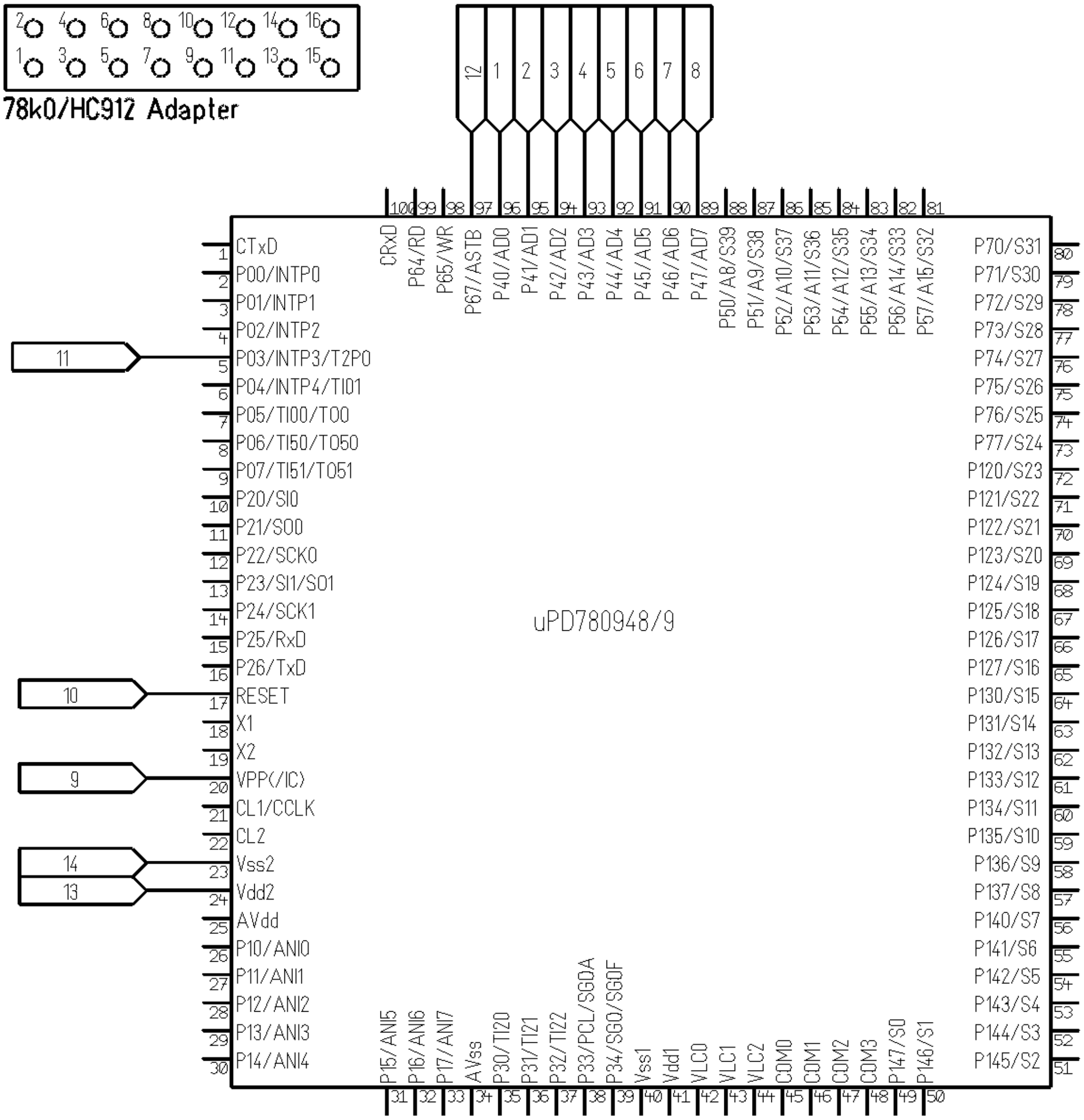
3.11.2 uPD780824/6/8A



3.11.3 uPD780973/4

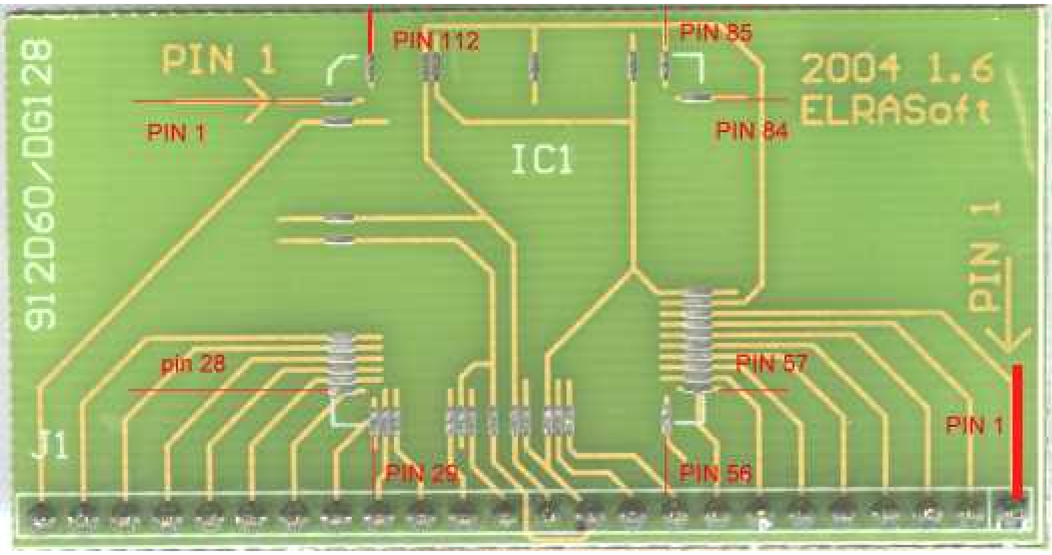


3.11.4 uPD780948/9



3.11.5 68HC912 QFP112

Desolder the MCU from the target board by hot air solder.
Solder the MCU on the 912D60/DG128 testing board according the picture below



Check for shorted pins by an ohm meter

Plug the 912D60/DG128 test board in the 78K0/HC912 adapter (J1 24 pins connector)

Look out for PIN1!

Plug the 78K0/HC912 Adapter in the UUSP - (DB9 Male Connector)

Connect the PC USB cable to the UUSP

Run UPA-USB Device Programmer Software and select a MCU - MC68HC912D60(A), MC68HC912DG128(A) or MC68HC912DC128A

Select a 4MHz Oscillator frequency (The 78K0/HC912 adapter use a 4MHz quartz)

Push the Read button, look at the red LED on the Adapter - It have to blink during reading (also during all other actions)

Blinking LED means that the MCU executes the code programmed in the external flash memory.

If in the future you'd like to access the MCU by BDM in circuit, push Disable BDM Lockout button. This will set NOBDM bit to 1 (Shadow word)

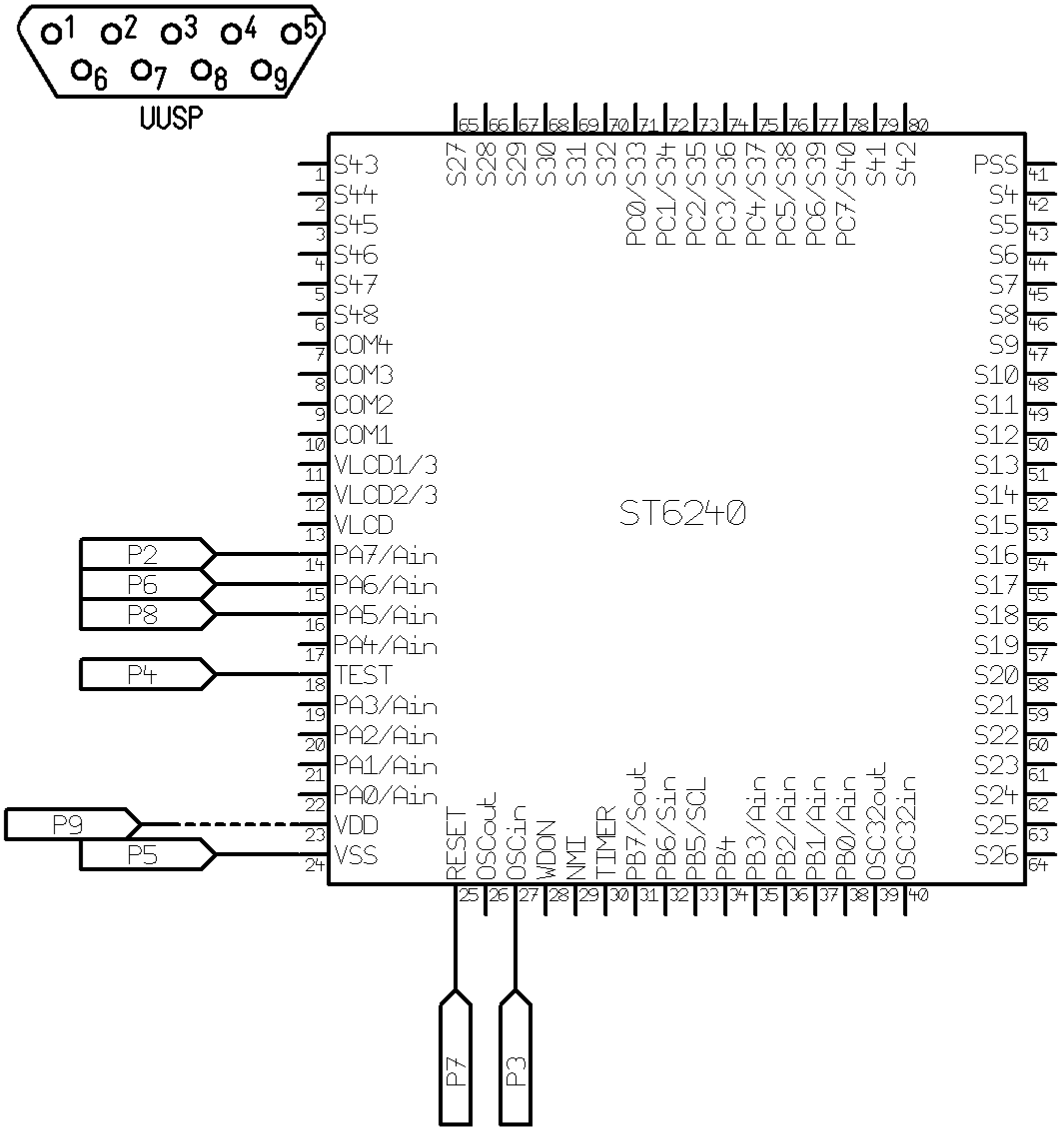
Note:

It's possible the original software (in the MCU internal flash) to enable BDM Lockout again after soldering of the MCU back on the target board.

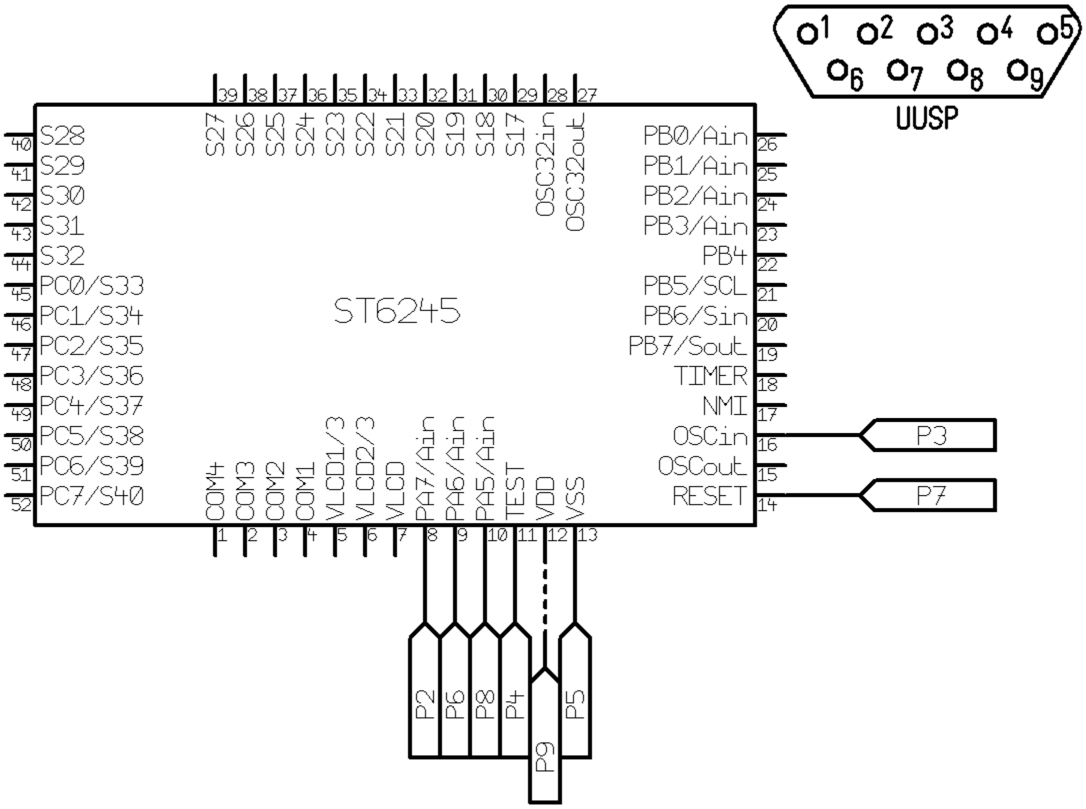
Such MCU can be read/program by BDM Lockout Adapter only

3.12 STMicroelectronics ST6

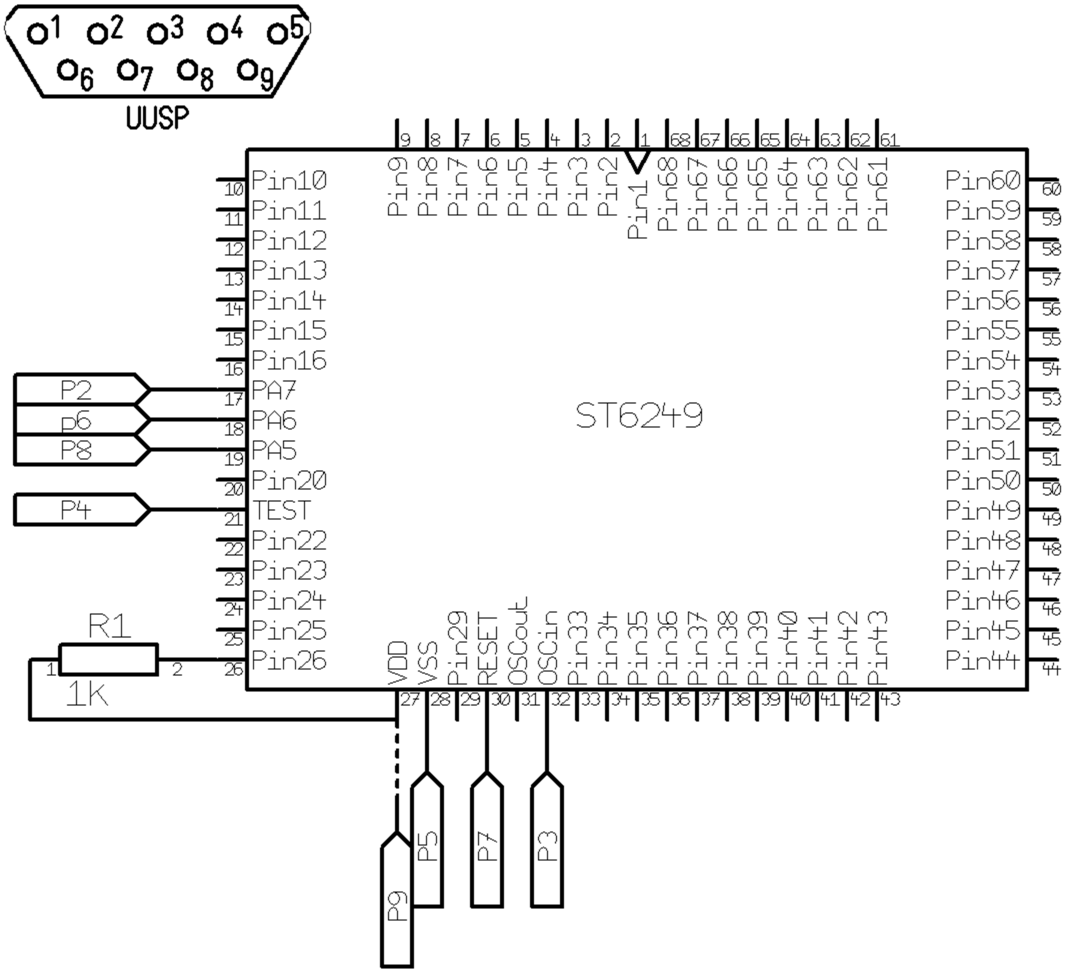
3.12.1 ST6240 QFP80



3.12.2 ST6245 QFP52



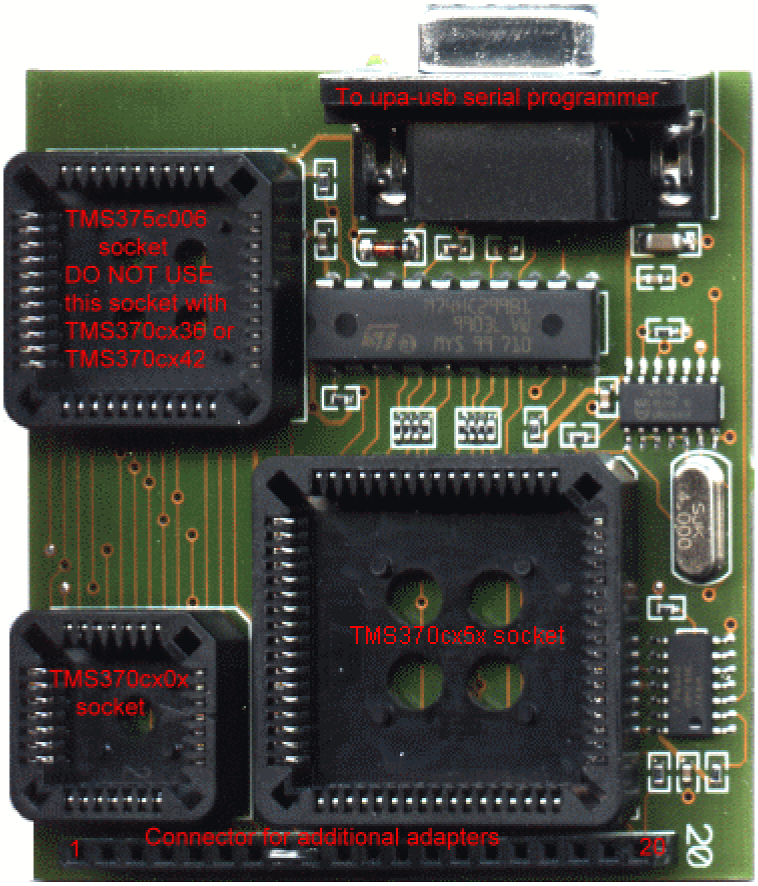
3.12.3 ST6249 QFP68



3.13 TMS Adapter

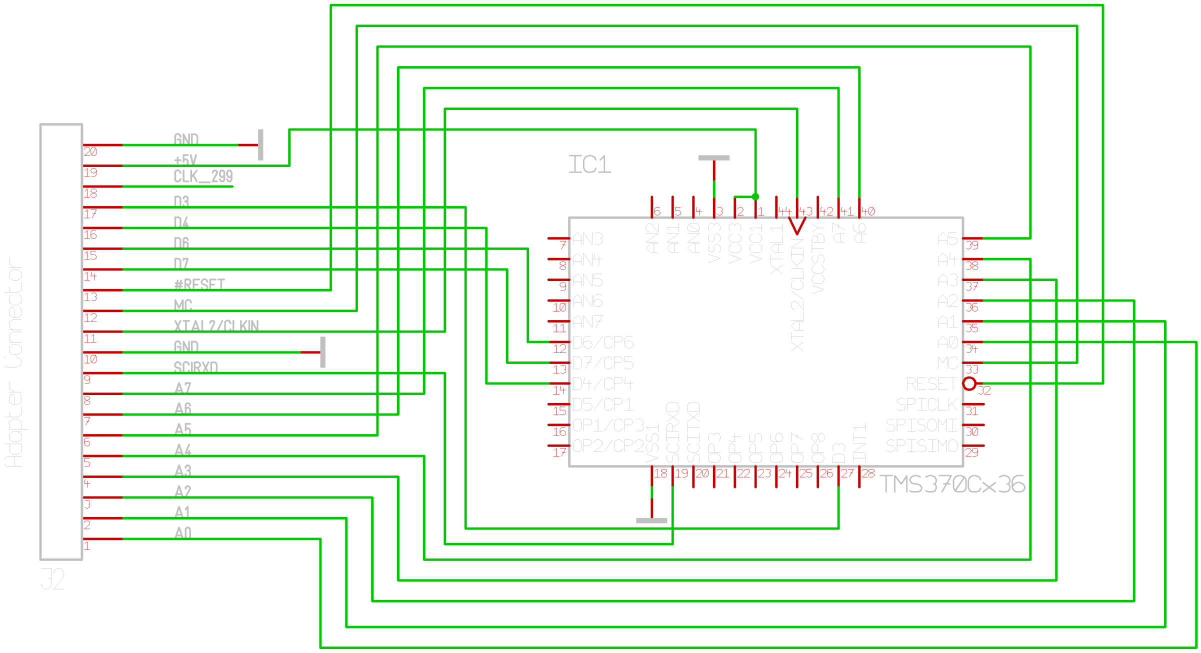
Supported Devices
TMS370cx0x, TMS370cx5x, TMS375c006
TMS370cx36 and TMS370cx42 by additional adapter

3.13.1 Socket Description

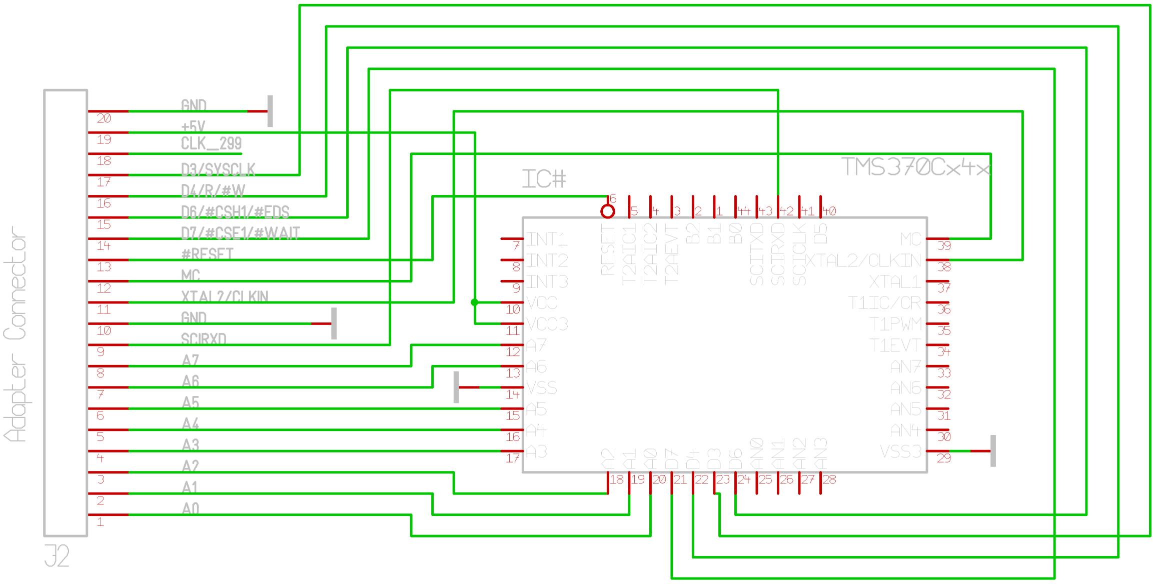

To upa-usb serial programmer
TMS375c006 socket DO NOT USE this socket with TMS370cx36 or TMS370cx42
TMS370cx0x socket
TMS370cx5x socket
Connector for additional adapters
1
20
20

3.13.2 Additional Adapter Schematics

3.13.2.1 TMS370cx36 Adapter Schematic



3.13.2.2 TMS370cx42 Adapter Schematic



4 Pascal Script Reference

4.1 Device Management

function AddAction(Caption, ProcName, DeviceName: **string**): **boolean**
function AddDevice(DeviceName, Description, GroupName, InheritedDeviceName: **string**): **boolean**
function AddDeviceGroup(GroupName, Description: **string**): **boolean**
function BlankCheckDevice: **boolean**
function GetDevice(DeviceName: **string**; **var** DevBase: TDevice): **boolean**
procedure HideDeviceOrGroup(DeviceOrGroupName: **string**)
function ProgramDevice: **boolean**
function ReadDevice: **boolean**
procedure ShowDeviceOrGroup(DeviceOrGroupName: **string**)
function VerifyDevice: **boolean**

4.1.1 AddAction

function AddAction(Caption, ProcName, DeviceName: **string**): **boolean**

4.1.2 AddDevice

function AddDevice(DeviceName, Description, GroupName, InheritedDeviceName: **string**): **boolean**

4.1.3 AddDeviceGroup

function AddDeviceGroup(GroupName, Description: **string**): **boolean**

4.1.4 BlankCheckDevice

function BlankCheckDevice: **boolean**

4.1.5 GetDevice

function GetDevice(DeviceName: **string**; **var** DevBase: TDevice): **boolean**

4.1.6 HideDeviceOrGroup

procedure HideDeviceOrGroup(DeviceOrGroupName: **string**)

4.1.7 ProgramDevice

function ProgramDevice: **boolean**

4.1.8 ReadDevice

function ReadDevice: **boolean**

4.1.9 ShowDeviceOrGroup

procedure ShowDeviceOrGroup(DeviceOrGroupName: **string**)

4.1.10 VerifyDevice

function VerifyDevice: **boolean**

4.2 File I/O

function AddOpenFileAction(Caption, DeviceName, FileName: **string**): **boolean**
function OpenFile(FileName: **string**): **boolean**

4.2.1 AddOpenFileAction

function AddOpenFileAction(Caption, DeviceName, FileName: **string**): **boolean**

4.2.2 OpenFile

function OpenFile(FileName: **string**): **boolean**

4.3 Hex Editor

function GetByteHexEdit(Offset: **integer**): **byte**
function GetSizeHexEdit: **integer**
procedure RefreshHexEdit
function SelectAllMemoryRange: **boolean**
function SelectEEPROMRange: **boolean**
procedure SetByteHexEdit(Offset: integer; Value: **byte**)
procedure SetProgramModifiedOnly(Value: **boolean**)
function SetProgramRange(StartVal, EndVal: **integer**): **boolean**

4.3.1 GetByteHexEdit

function GetByteHexEdit(Offset: **integer**): **byte**

4.3.2 GetSizeHexEdit

function GetSizeHexEdit: **integer**

4.3.3 RefreshHexEdit

procedure RefreshHexEdit

4.3.4 SelectAllMemoryRange

function SelectAllMemoryRange: **boolean**

4.3.5 SelectEEPROMRange

function SelectEEPROMRange: **boolean**

4.3.6 SetByteHexEdit

procedure SetByteHexEdit(Offset: **integer**; Value: **byte**)

4.3.7 SetProgramModifiedOnly

procedure SetProgramModifiedOnly(Value: **boolean**)

4.3.8 SetProgramRange

function SetProgramRange(StartVal, EndVal: **integer**): **boolean**

4.4 Message and Input Boxes

procedure AddMsg(Text: **string**)
procedure ClearMsgs
function InBox(Caption, EditLabel: **string**; var Value: **string**): **boolean**
function MsgBox(Text, Caption: **string**; Flags: **integer**): **integer**

4.4.1 AddMsg

procedure AddMsg(Text: **string**)

4.4.2 ClearMsg

procedure ClearMsgs;

4.4.3 InBox

function InBox(Caption, EditLabel: **string**; var Value: **string**): **boolean**
Displays a prompt in a dialog box, waits for the user to input text or click a button, and returns the contents of the text box to Value parameter.

Parameters
Caption: ***string***
string that contains the input box title
EditLabel: ***string***
string that contains the edit control label
Value: **string**

Return Value
If the user clicks OK or presses ENTER, the InBox function returns True and Value parameter, whatever is in the text box. If the user clicks Cancel, the function returns False.

4.4.4 MsgBox

function MsgBox(Text, Caption: **string**; Flags: **integer**): **integer**
The MsgBox function creates, displays, and operates a message box

Parameters
Text: ***string***
string that contains the message to be displayed
Caption: ***string***
string that contains the dialog box title
Flags: ***integer***
Specifies the contents and behavior of the dialog box. This parameter can be a combination of flags from the following groups of flags. To indicate the buttons displayed in the message box, specify one of the following values.

MB_OK
The message box contains one push button: **OK**. This is the default.
MB_OKCANCEL
The message box contains two push buttons: **OK** and **Cancel**.
MB_ABORTRETRYIGNORE
The message box contains three push buttons: **Abort**, **Retry**, and **Ignore**.

MB_YESNOCANCEL
The message box contains three push buttons: **Yes**, **No**, and **Cancel**
MB_YESNO
The message box contains two push buttons: **Yes** and **No**.
MB_RETRYCANCEL
The message box contains two push buttons: **Retry** and **Cancel**.

To display an icon in the message box, specify one of the following values.
MB_ICONHAND
A stop-sign icon appears in the message box.
MB_ICONQUESTION
A question-mark icon appears in the message box.
MB_ICONEXCLAMATION
An exclamation-point icon appears in the message box.
MB_ICONASTERISK
An icon consisting of a lowercase letter *i* in a circle appears in the message box.
MB_ICONWARNING
An exclamation-point icon appears in the message box.
MB_ICONERROR
A stop-sign icon appears in the message box.
MB_ICONINFORMATION
An icon consisting of a lowercase letter *i* in a circle appears in the message box.
MB_ICONSTOP
A stop-sign icon appears in the message box.

Return Value
If the function fails, the return value is zero.
If the function succeeds, the return value is one of the following menu-item values.

IDABORT	Abort button was selected.
IDCANCEL	Cancel button was selected.
IDIGNORE	Ignore button was selected.
IDNO	No button was selected.
IDOK	OK button was selected.
IDRETRY	Retry button was selected.
IDYES	Yes button was selected.

4.5 Miscellaneous

Application: TApplication
InputForm: TForm

function IntToHex(Value: **Integer**; Digits: **Integer**): **string**
procedure SetProductInfo(ProductName, Description: **string**)

4.5.1 Application

Application: TApplication

4.5.2 InputForm

InputForm: TForm

4.5.3 IntToHex

function IntToHex(Value: **Integer**; Digits: **Integer**): **string**

4.5.4 SetProductInfo

procedure SetProductInfo(ProductName, Description: **string**)

4.6 RemObjects Pascal Script

Types
Reserved words
Statements
Library

4.6.1 Library

function FloatToStr(e: extended): **string**
function IntToStr(i: Longint): **string**
function StrToInt(s: **string**): Longint
function StrToIntDef(s: **string**; def: Longint): Longint
function Copy(s: **string**; ifrom, icount: Longint): **string**
function Pos(substr, s: **string**): Longint
procedure Delete(**var** s: **string**; ifrom, icount: Longint): **string**
procedure Insert(s: **string**; **var** s2: **string**; ipos: Longint): **string**
function GetArraylength(**var** v: **array**): Integer
procedure SetArrayLength(**var** v: **array**; i: Integer)
function StrGet(S : **String**; I : Integer) : Char
function StrSet(c : Char; I : Integer; **var** s : **String**) : Char
function Uppercase(s : **string**) : **string**
function Lowercase(s : **string**) : **string**
function Trim(s : **string**) : **string**
function Length(s : **String**) : Longint
procedure SetLength(**var** S: **String**; L: Longint)
function Sin(e : Extended) : Extended
function Cos(e : Extended) : Extended
function Sqrt(e : Extended) : Extended
function Round(e : Extended) : Longint
function Trunc(e : Extended) : Longint
function Int(e : Extended) : Longint
function Pi : Extended
function Abs(e : Extended) : Extended
function StrToFloat(s: **string**): Extended
function FloatToStr(e : Extended) : **String**
function Padl(s : **string**; I : longInt) : **string**
function Padr(s : **string**; I : longInt) : **string**
function Padz(s : **string**; I : longInt) : **string**
function Replicate(c : char; I : longInt) : **string**
function StringOfChar(c : char; I : longInt) : **string**

4.6.2 Reserved words

AND
ARRAY
AS
BEGIN
CASE
CHR
CLASS
CONST
CONSTRUCTOR
DESTRUCTOR
DIV
DO
DOWNTO
ELSE
END
EXCEPT
EXIT
EXPORT
EXTERNAL

FINALLY
FOR
FORWARD
FUNCTION
GOTO
IF
IMPLEMENTATION
IN
INHERITED
INTERFACE
IS
LABEL
MOD
NIL
NOT
OF
OR
ORD
OUT
OVERRIDE
DEFAULT
PRIVATE
PROCEDURE
PROGRAM
PROPERTY
PROTECTED
PUBLIC
PUBLISHED
RECORD
REPEAT
SET
SHL
SHR
THEN
TO
TRY
TYPE
UNIT
UNTIL
USES
VAR
VIRTUAL
WHILE
WITH
XOR

4.6.3 Statements

begin statement1; ... statementN; **end**
if expression **then** statement1 **else** statement2
for counter := expression1 **to|downto** expression1 **do** statement
case expression **of** caseList1: statement1; ... caseListn: statementN; **end**
repeat statement **until** expression
while expression **do** statement
with object **do** statement
uses

try statement **except|finally** statement **end**
exit
continue
break

4.6.4 Types

Byte, Shortint, Word, SmallInt, Cardinal, Longint, Integer
Char
String
Real, Double, Single, Extended,Comp
Boolean
Array
Record
Variant
Enumerations
Classes