CPU Module EMUL12-PC/CPU-MCS912DP256-xx for the Motorola HCS12 Microcontrollers

Operating Instructions

by Robert Boys **V2.6 January 17, 2002**

This document provides jumper information on the DP256 personality card for the Nohau EMUL-S12D full emulator. This emulator supports all Motorola HCS12 processors and this personality card supports all MC9S12D parts including the MC9S12DP256. The part number for the module is EMUL12-PC/CPU-MCS912DP256-16 or -25. The circuit diagrams enclosed are

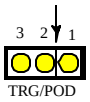
1) The emulator will work stand-alone (no target connected) with the default settings.

2) These jumpers are named where possible to conform to the Motorola data sheet.

3) The jumpers are sorted into separate clock and voltage sections in this note.

4) "PLL" as in EXTAL/PLL denotes the Nohau internal PLL clock source and not the Motorola on-chip PLL clock. The Nohau PLL supplies a 2.5 VPP signal and the frequency is set in the Seehau software under the Config, Emulator window. By default the Nohau PLL source is used to supply EXTAL without the Motorola PLL activated and the resulting E clock or bus speed frequency will be 0.5 that is entered into the Seehau software.

5) The jumpers have a pin 1 which is indicated on the board silkscreen as a white line as indicated by the arrow on this sample. Pin 1 is also shown on the schematics. Jumpers can be set by the numbers or with the setting indicated on the silkscreen i.e. XTL/PLL or TRG/POD.

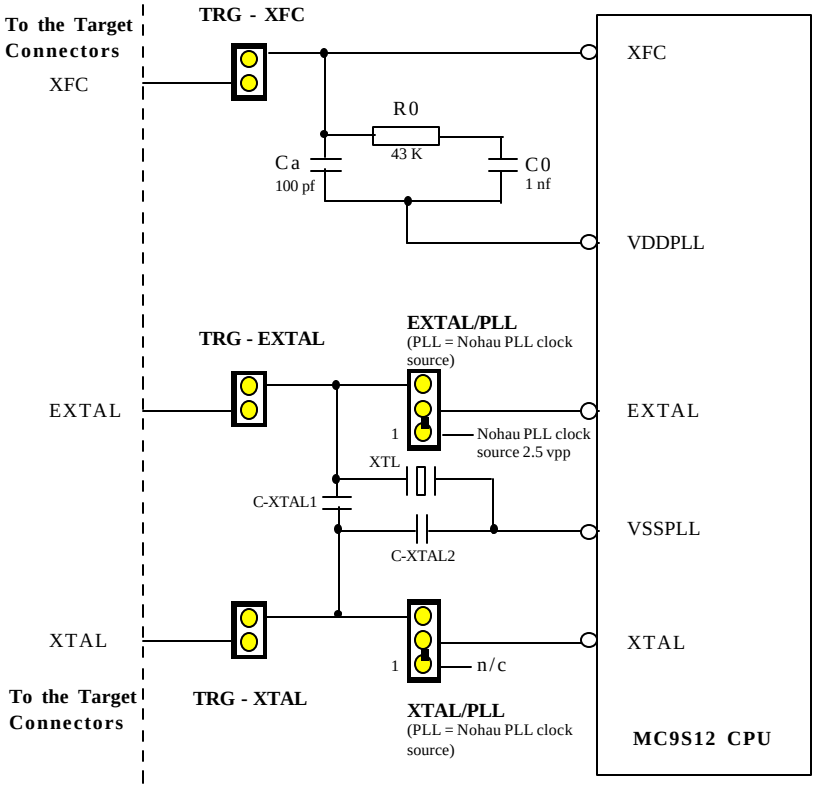


6) TRG means Target and the signal will connect to the target system if set to TRG or to the emulator hardware if set to the POD position. The POD position is normally jumper pins 1 and 2. The factory set default position is shown by a black bar such as seen on the jumper here. 

7) The gold header pins accessible on the top of this personality module connect directly to the target connectors and are labelled. These connectors do not necessarily connect directly to the appropriate pin on the DP256 controller. Recreated port pins will go through emulator circuitry and hence are not directly connected. The recreated ports are A, B, E and K and some additional control signals and are not directly connected. Peripheral modules on other ports are generally directly connected. Modules such as CAN, SPI and A/D are not affected or controlled by the emulator logic.

The on-board microcontroller is always operated in expanded wide mode even if another mode such as single-chip is being specified by the user. The pins on the on-board chip will show this expanded mode while the header pins will look like the user specified mode. Do not take signal tests directly from the chip as you may receive erroneous results.

Clock Jumpers



1) This set of jumpers is used to select either the Nohau PLL clock source or a user supplied crystal or external source is used to supply a clock signal to the emulation CPU on the emulator. In addition, jumpers are used to connect the XFC, EXTAL or XTAL pins of the CPU to the target board as desired by the user.

2) The default values of C0 is 1 nf (1000 pf) and Ca is 100 pf. R0 is 43 Kohm. These are the values as installed by Nohau. Contact your Motorola representative for the best values appropriate for your frequency of operation or go to www.nohau.com/s12calc/pll_302.html for a circuit calculator supplied by Motorola.

The TRG - XFC jumper s elects the PLL filtering components from the target system. The Nohau installed C0, R0 and Ca must be removed if this jumper is installed.

3) "PLL" as in EXTAL/PLL denotes the Nohau internal PLL clock source and not the Motorola on-chip PLL clock. The Nohau PLL supplies a 2.5 VPP signal and the frequency is set in the Seehau software under the Config, Emulator window.

4) Nohau PLL clock source can be used to supply either the Motorola divide-by-2 clock or the Motorola PLL circuitry as determined by the setting of the PLLSEL bit in the CLKSEL register. If the Nohau PLL source is used to supply EXTAL without the Motorola PLL, the E clock or bus speed frequency will be 1/2 of the value entered into the Seehau software.

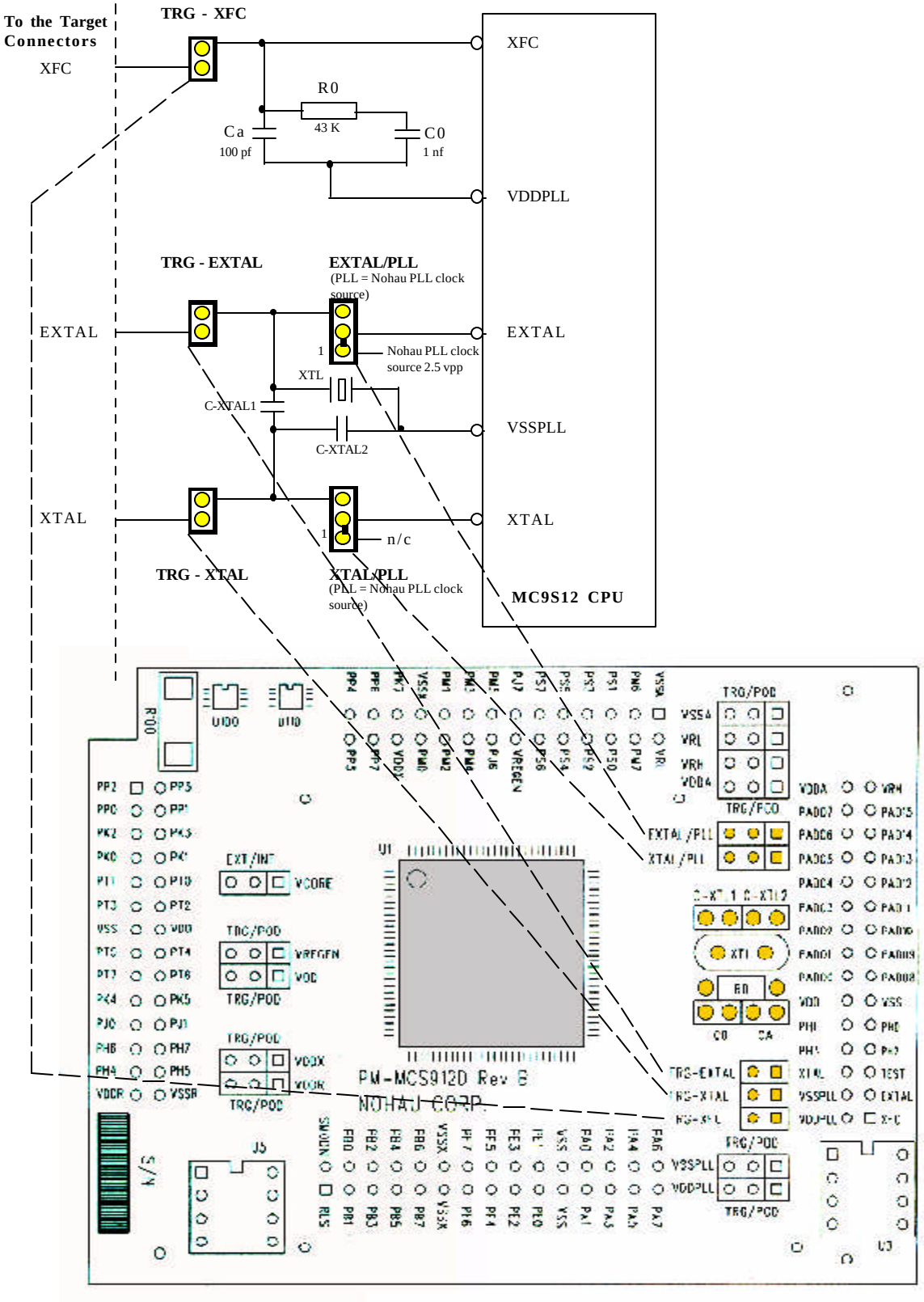
If the Motorola PLL is used, then the output frequency will be determined by the ratio of the SYNR and REFDV registers. In this case the Nohau PLL will be OSCCLK in the Motorola PLL circuitry. The jumper EXTAL/PLL supplies the Nohau PLL clock source to the EXTAL pin of the CPU.

The Nohau PLL Clock Source and How It Works

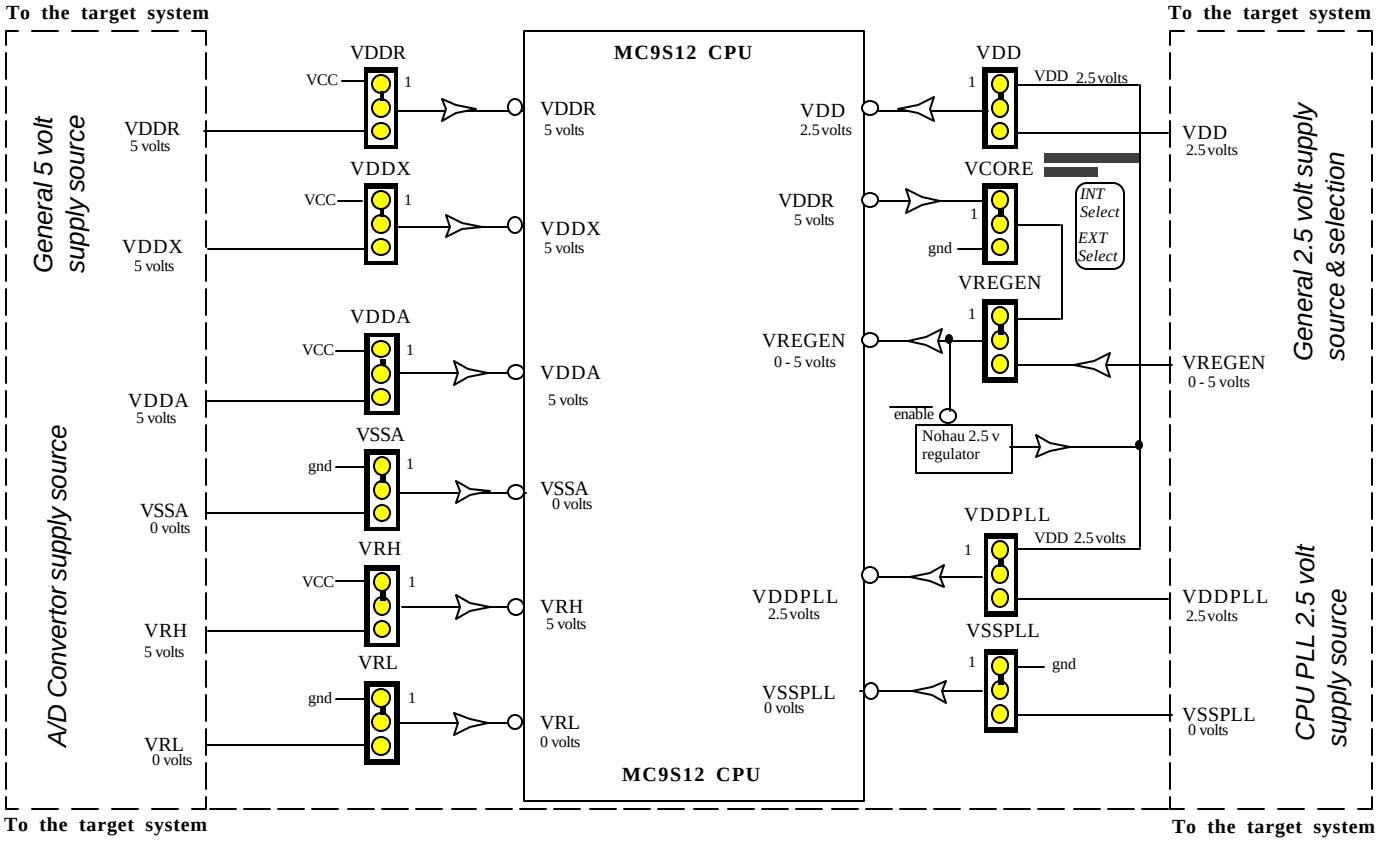
The Nohau PLL clock source frequency is selected in the Seehau software under Config, Emulator. The value entered in the box titled Clock (MHz) and can range from 3.125 MHz to the maximum frequency of the emulator which can be 16, 24 or 25 MHz. These values are the possible outputs of the PLL circuitry but may or may not be appropriate for your system configuration. Practical frequency settings could be narrower than these maximum ranges.

The value entered has a software resolution of 15 digits but this exceeds the capabilities of the PLL circuitry. The PLL will provide a frequency within +/- .05% of the value entered. Seehau will return to the box the frequency it was able to select. If 3.125678 is entered and APPLY is clicked, 3.12568 is returned in the box and this signal is applied to the emulator circuitry and pin 1 of jumper EXTAL/PLL. This clock is 2.5 Volts PP.

Clock Jumpers to CPU Module Map



Voltage Jumpers



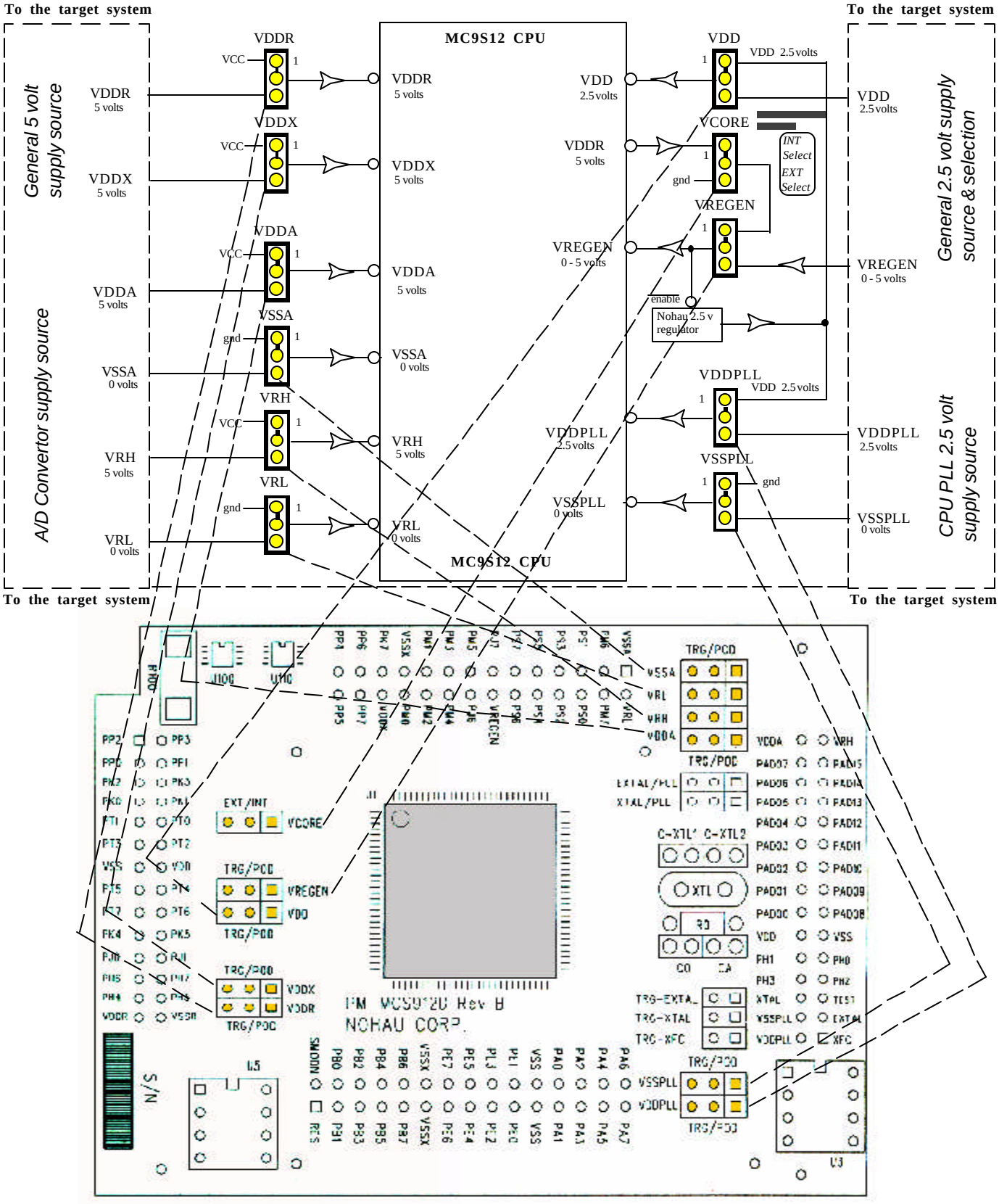
- "VCC", "VDD", "gnd" are supplied by the Nohau emulator POD. VCC is 5 and VDD is 2.5 volts.
- Upper positions are POD, lower are TRG (target). These are marked on the CPU module. VCORE is the exception and is marked accordingly on this document and on the module.
- ▷ indicates the direction of the effect, not current or signal electron flow.
- VDD is the 2.5 volt output of the CPU internal voltage regulator. Do not connect 5 volts to this point or the CPU will likely be destroyed. This is also true for VDDPLL.

All names in **BOLD** are jumpers and not CPU pins.

VCORE selects if the CPU internal 2.5V regulator will be turned on or off when the **VREGEN** jumper is in the POD position (default). If **VREGEN** jumper is in the TRG position, the target will determine the state of the internal regulator. If the CPU internal regulator is turned off, the Nohau external 2.5V regulator on the CPU-Module is automatically activated. This isused to power the VDD & VDDPLL pins of the CPU as selected by the **VDD** & **VDDPLL** jumpers (POD) or the target system if set to the TRG position.

The other jumpers determine whether voltage sources or grounds are supplied by the target system or the emulator system. The default position is to select the emulator sources.

Voltage Jumpers to CPU Module Map



Getting the Motorola PLL to Work

A) The Basics

The emulator as it is delivered from Nohau with the default jumper settings can be run using either the Motorola divide-by-2 clock or the Motorola PLL mechanism. Recall that the word "PLL" as indicated on the personality module or in this document refers to the Nohau PLL clock generator and not the Motorola CPU PLL. We will refer to the Motorola PLL with "CPU PLL". Consult the Motorola datasheet for more information on their Clock and Reset Generator (CRG) module.

The default is the divide-by-2 clock and you can switch it to CPU PLL by programming the SYNR and REFDV registers then activating the CPU PLL by setting the PLLSEL bit in the CLKSEL register. No jumper changes are necessary.

You can switch the emulator from CPU PLL to divide-by-2 with your code running on the emulator or by typing in the appropriate values into the specified register windows in Seehau while the emulator is in monitor mode (i.e. emulation is stopped).

B) Setting Up the Emulator

These instructions will construct the windows in the Seehau software to allow you to manipulate the CPU PLL circuitry. You will need an oscilloscope to view the oscillator waveform and measure its frequency.

1) Start Seehau in the normal fashion and right mouse click on a register window. The default Reg_1 window will do fine.
2) Select Add Special Register (SFR) and double click or press Enter. The SFR window as shown in Figure 1 opens up.

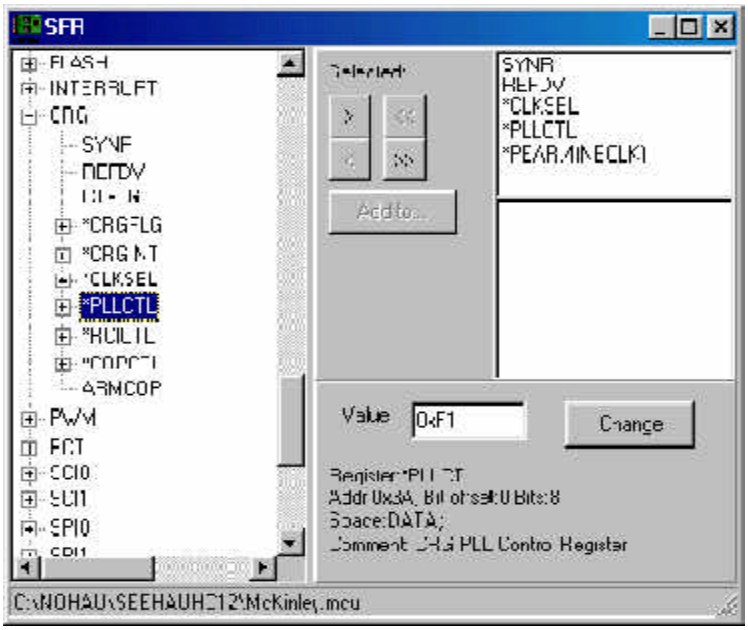


Figure 1

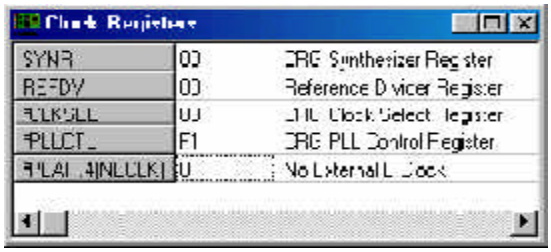


Figure 2

3) Double click on CRG or click on the "+" mark beside it. You might have to scroll down to view this.
4) In turn, highlight and click on the ">" symbol to place the register names in the window on the right as shown in Figure 1. You will find the NECLK=0 under I/O Control, *Pear. NECLK is a bit in the PEAR register and is used to send the E Clock to the PE4 header pin on the top of the CPU module for easy access by an oscilloscope.
5) Click on Add To.. and select New Window. A window similar to Figure 2 will open up. It will probably be labelled Reg_2.
6) Right click on this new register window and select Show Description.
7) Right click on this register window again and select Change Caption. I entered Clock Registers and Figure 2 is the result.
8) In the main Seehau screen, select Config, and select Save Settings to save the entire workspace. Give it a name of your choice or use the default startup.bas. You can also select Save Window to save the register window if it is currently in focus to a macro that can be easily recalled with a button. See the Getting Started manual for more information.

Getting the Motorola PLL to Work *continued...*

C) Getting the Hardware Configured

1) Have Seehau operating but not running any user programs. The green GO icon should be visible.

2) Connect an oscilloscope to the header pin PE4 on the top of the Nohau personality board. This pin is readily accessible.

3) In the register window Figure 2, enter a 0 into Pear.4 (NECLK) and press Enter. This sends the E clock signal to PE4. This will now be visible on the oscilloscope. Entering a 1 will turn it off. In Single Chip mode, this bit defaults to a 1 (click on the Nohau RESET icon) and in Expanded modes it defaults to 0. We are using Single Chip but you can use Expanded if you prefer.

4) Select the Pear.4 (NECLK) line and right click on it. Select Change Attributes and check the Enable Reset Value. The default value of 0x0000 is already entered. Click on OK. Every time you click on the Nohau RESET icon, the Pear.4(NECLK) bit will be set to zero.

   You can do this later to the SYNR and REFDV registers with your own default values. You must resave the settings under Config, Save Settings in order for these effects to be activated the next time you open Seehau.

5) Click on Config, Emulator in the main Seehau screen and note the value in the Clock(MHz) box. The default is 16 MHz and represents the value of the Nohau PLL clock source as fed through the EXTAL/PLL jumper to the EXTAL pin on the DP256 microcontroller.

   The E clock frequency in the divide-by-2 mode (no CPU PLL) will be 1/2 the Nohau PLL frequency. The default E clock frequency displayed on your oscilloscope will be 8 MHz with and EXTAL frequency of 16 MHz.

   The Nohau PLL or EXTAL frequency is labelled OSCCLK in the Motorola PLL specifications. It is readily accessible for monitoring on the EXTAL/PLL jumper pin 1. It is a 2.5 volt PP signal to correspond with the CPU core voltage. You can easily change this frequency by typing in a new value and pressing Enter or Apply. You can enter values from 3.125 KHz to 16, 24 or 25 MHz depending on the maximum frequency of the emulator. See page 2 for more Nohau PLL information.

D) Basic Motorola PLL Information

1) The Motorola PLL is selected by entering 80 in the CLKSEL register in the register window of Figure 2. A zero turns it off. This bit actually switches from the divide-by-2 circuitry to the CPU PLL. When the CPU PLL is off, the E clock will be OSCLK divided by 2.

2) The frequency is determined by the formulae: $PLLCLK = 2 \times OSCCLK \frac{SYNR + 1}{REFDV + 1}$ or $ECLK = OSCCLK \frac{SYNR + 1}{REFDV + 1}$

3) PLLCLK does not show up externally on the CPU.

4) E clock is PLLCLK / 2 and is visible to the outside world on PE4 as described in 3) above.

5) The minimum OSCLK frequency is from 1 MHz to the limit of the emulator whether it is 32, 48 or 50 MHz.

6) OSCLK is the oscillator frequency as fed into the CPU EXTAL pin via the EXTAL/PLL jumper.

7) OSCLK is the same frequency as entered in the Seehau software in the Config, Emulator dialog window.

8) E Clock is derived from the OSCLK and is 1/2 the frequency of OSCLK when in divide-by-2 mode. In CPU PLL mode it will be the ratio of SYNR+1 and REFDV+1.

9) Register and frequency v alues used must be in appropriate ranges. It is possible to set the CPU PLL to provide frequencies outside the range of the CPU and/or the emulator. Erroneous microcontroller or emulator operation will result.

10) The CPU PLL filter values as determined by C0, Ca and R0 must be selected according to the frequencies synthesized. Contact your Motorola representative for these values.

11) We will only program and pay attention to the SYNR, REFDV registers and the PLLSEL bit in the CLKSEL register for this demonstration. This is to demonstrate the Motorola PLL with the Nohau emulator functions as described and provides a good reference point for your software design. Consult the Motorola datasheet for information on other clock registers or go to www.nohau.com/s12calc/pll_302.html for a circuit calculator supplied by Motorola. The Nohau emulator supports all the clock registers even if they are not mentioned here.

E) Basic Nohau Emulator Information

1) You can change the CPU PLL registers on-the-fly while running your program by entering the values into a properly set data window. The Motorola BDM is used to update these values and is described on the next page.

2) You can change them when the emulator is stopped by entering new values into a register window.

3) You can also change them in your program code. An example is given later in this document.

4) You may change your CPU PLL frequency as often as you want and the full Nohau emulator will track these changes. It will make any modifications to its operating environment as needed and in real time without stealing machine cycles.

5) There is an issue of the LOCK bit not correctly indicating the CPU PLL is out of lock after writing to one of the PLL registers. There is a simple workaround which consists of a short time delay before you test the LOCK bit. This is an issue only when you are switching register values with your own code. It is not an issue if you are entering register values into the emulator manually because of the inherent time delay.

F) Programming Methods for the Motorola PLL Registers

The standard method of programming the CPU PLL registers (see Figure 2) for the purposes of this article is the following:

1) Enter an appropriate value into the Seehau software for OSCLK. 8 MHz is used in this application note.

2) Make sure the CLKSEL register contains 0. This turns the CPU PLL off by setting bit PLLSEL off.

3) Program the appropriate values into SYNR and REFDV. We will use in this note values will be in the area of 1, 2, 3 or 4.

4) Enter 80 into CLKSEL. This sets bit PLLSEL on. If the values above are in range, the E Clock will change to the new frequency.

5) To change the CPU frequency again, simply write a 0 to CLKSEL, enter new values to SYNR and REFDV as needed and write an 80 to CLKSEL to activate the CPU PLL again.

G) Changing the CPU PLL via the Seehau Register Window (see Figure 2)

1) Make sure the emulator is setup as described previously under *Getting the Hardware Configured.*

2) Enter 8 into the Clock MHz box under Config, Emulator in the main Seehau window. This sets OSCLK to 8 MHz. The frequency as measured on ECLK (PE4) will be 4 MHz if the CPU is in divide-by-2 mode.

3) Enter 0 into the Pear 4 NECLK register. This sends the E clock to Port E.4. (PE4)

4) Enter 0 into the CLKSEL register and either press Enter or click on another register. CPU PLL is now ensured to be off.

5) Enter 1 into SYNR and 3 into REFDV. The frequency of PLLCLK will be (2)(8)(1+1)/(3+1) = 8 MHz. ECLK will be 4 MHz.

6) Enter 80 into CLKSEL and press Enter. 4 MHz will be available on the scope which is the same as before.

7) To test if the CPU is in PLL mode, grab the PLL filter components with your fingers and an increased fuzziness in the scope display will result. If the PLL filter components are very close to certain values, this effect may be very small. If the CPU is in divide-by-2 mode, doing this will have reduced or no effect. This is called the SRTest after its discoverer.

8) Enter 0 into the CLKSEL register and press Enter to disable the CPU PLL. The SRTest will fail. The CPU PLL is off.

9) Enter 4 into REFDV and then 80 into CLKSEL and press Enter. ECLK will now equal (8)(1+1)/(4+1) = 3.2 MHz. Confirm this with your oscilloscope. Try the SRTest. It will indicate the CPU PLL is running properly.

10) Enter 0 into the CLKSEL register and press Enter to disable the CPU PLL. The SRTest will fail.

11) Enter 3 into SYNR and then 80 into CLKSEL and press Enter. ECLK will now equal (8)(3+1)/(4+1) = 6.4 MHz.

12) Reset the emulator by clicking on the RESET icon.

If the waveforms become fuzzy, this likely is the result of the PLL filter components C0, Ca and R0 not being correct for this frequency of operation. Consult Motorola for correct values or go to www.nohau.com/s12calc/pll_302.html for a circuit calculator supplied by Motorola. This example worked with the Nohau default values.

H) Software Control of the Motorola PLL

The Motorola PLL can be controlled directly by software and switching on-the-fly is possible. There are a few issues with the main one being the LOCK bit problem. The LOCK bit (bit 3 in the CRGFLG register) indicates that the PLL is within a certain range of the desired frequency and is used to detect if it is allowable to enable the CPU PLL so it will become the system clock.

The problem is the setting and clearing the LOCK bit is delayed after the SYNR and REFDV is written to. Normally the SYNR, REFDV and perhaps other registers will be initialized by your code and then the PLLSEL bit is set to enable the CPU PLL. The code must wait until the LOCK bit is set indicating the PLL is stable before it can be used in the system and before program execution continues. Normally this would be accomplished by testing the state of the LOCK bit before setting PLLSEL.

The mentioned delay will result in the LOCK bit still asserted even if the PLL is not locked if tested immediately after setting the SYNR, REFDV or PLLCTL. A successful workaround is to insert a small time delayafter configuring SYNR, REFDV or PLLCTL but before setting the PLLSEL bit. The LOCK bit can be tested and when it indicates the CPU PLL is stable and locked, it is safe to enable it. Assembly code and C source examples are given.

I) Nohau Test Program: Switches the CPU PLL On and Off Continuously

A Nohau test program that switches the CPU PLL clock on and off is included in the Seehau software release. It is in the \Examples\DP256\Tstpll directory in the installed software. The asm code is commented.

It loads to address 400H so you need to have the EEPROM turned off to run it (check the Disable EEPROM check-box).

Open a RUN-TIME-DATA window to point to 7F0H. This shows whether continuance BDM communication is maintained during the rapid changes in the BDM communication rate when the PLL is selected and deselected.

Enter 8 MHz clock speed into the Config, Emulator window which then creates 4 MHz & 16 MHz bus speeds at Non-PLL and PLL modes accordingly.

The program selects and deselects the PLL to be used as the system clock many times in an endless loop. It has a ratio of 4:1 between PLL mode to non-PLL mode. It increments two counters at 7F0H & 7F8H every time PLL or Non-PLL mode becomes active. This will be visible in the data window.

J) Motorola Assembly Example

This example is taken from a application note by Gordon Doughman. Go to www.motorola.com/mcu and search for AN2153.

“Shown below is a code snippet from one of my up coming app notes. To obtain 25 MHz operation the reference clock must be an integer multiple of 25 MHz. So, if you are using the 16 MHz oscillator on the evaluation board, the value for OscClk would be changed to 16000000. The value for fEclock would be changed to 25000000. The only way to obtain a 25 MHz E-clock with a 16 MHz oscillator is to have a reference frequency of 1 MHz, so RefClock should be set to 1000000. For 25 MHz operation, a 5 MHz crystal/oscillator would be a better choice.”

```
OscClk:         equ     8000000                         ;    oscillator    clock    frequency.
fEclock:        equ     24000000                        ;    final    E-clock    frequency    (PLL).
RefClock:       equ     8000000                         ;   reference   clock   used   by   the   PLL.
REFDVVal:       equ         (OscClk/RefClock)-1         ;   value   for   the   REFDV   register.
SYNRVal:        equ          (fEclock/RefClock)-1       ;   value   for   the   SYNR   register.
    if    OscClk>12800000
FCLKDIVVal:     equ           (OscClk/200000/8)+FDIV8   ;   value   for   the   FCLKDIV   register.
   else
FCLKDIVVal:     equ        (OscClk/200000)              ;   value   for   the   FCLKDIV   register.
    endif

   ldab        #REFDVVal                                ;   set   the   REFDV   register.
   stab        REFDV
   ldab        #SYNRVal                                 ;   set   the   SYNR   register.
   stab        SYNR
   nop                                                  ;   nops   required   for   bug   in   initial   silicon.
   nop
   nop
   nop
    brclr      CRGFLG,#LOCK,*                           ;   wait   here   till   the   PLL   is   locked.
    bset       CLKSEL,#PLLSEL                           ;   switch   the   bus   clock   to   the   PLL.
```

K) C Code Example

The **FOR loop** for (i =0 ; i <4 ; i++); provides a delay to allow the LOCK bit to be cleared so testing it will make sense. Remove this line and the program will not work correctly.

```
CLKSEL.Data.Byte      &=      ~PLLSEL;        //    disable    PLL
SYNR.Data.Byte     =     0x3;
REFDV.Data.Byte     =     0x01;
for   (i   =0   ;   i   <4   ;   i++);        //    small    delay
while           (!(CRGFLG.Data.Bit.B3))
CLKSEL.Data.Byte      =      PLLSEL;          //    enable    PLL
```


Nohau **Tel: (888) 886-6428**
275 E. Hacienda Avenue Tel: (408) 866-1820
Campbell, California 95008 Fax: (408) 378-7869
Email: sales@nohau.com Web: www.nohau.com

To locate your local representative go to www.nohau.com/reps

Enable Engineering **Tel: (800) 68-NOHAU**
430 Peninsula Ave #14 Tel: (650) 375-0409
San Mateo, CA 94401 Fax: (650) 375-8666
Email: sales@eecosales.com Web:ww.eecosales.com

Dearborn Group **Tel: (248) 488-2080**
27007 Hills Tech Ct.
Farmington Hills, MI 48331 Fax: (248) 488-2082
Email: dg@dgtech.com Web: www.dgtech.com

Nohau UK Ltd. **Tel: +44 1962 733 140**
The Station Mill Fax: +44 1962 735 408
Alresford, Hampshire S024 9JG
Email: sales@nohau.co.uk Web: www.nohau.co.uk

MICROTASK Embedded S.r.l. **Tel: +39 02 6680 2557**
Via Paolo Bassi, 1 Fax: +39 02 6900 8174
20159 Milano
Email: mcavallaro@microtask-embedded.it
Web: www.microtask-embedded.it

Nohau Elektronik GmbH **Tel: +49-7041-81803-0**
Bahnhofstrasse 52 Fax: +49-7041-81 803-18
D-75417 Muhlacker, Germany
Email: info@nohau.de Web: www.nohau.de

Nohau Elektronik AB **Tel: +46 40 59 22 00**
Derbyvägen 4 Fax: +46 40 59 22 29
S-212 35 Malmö, Sweden
Email: info@nohau.se Web: www.nohau.se

Emulations S.A.R.L. **Tel: +33 1 69 41 28 01**
A13 - Burospace Fax: +33 1 60 19 29 50
Chemin de Gizy
91572 Bievres Cedex, France
Email: vlm@emulations.fr Web: www.emulations.fr

Microcomputer Solutions Pvt. Ltd.
Tel: +91 20 4210615/4217725
22/6, Premnagar Fax: +91 20 4216798
Pune 411 037, India
Email: mspls@vsnl.com Web: www.mspls.com

Sophia Systems Co., Ltd. **Tel: +81 44 989 7239**
6-2, Minamikurokawa, Asao-ku Fax: +81 44 989 7005
Kawasaki-city, Kanagawa 215-8588, Japan
Email: TSC@sophia-systems.co.jp
Web: www.sophia-systems.co.jp