



AN-ELNEC-EN-ISP-HCS12

Application note for In-System Programming of Motorola/Freescale HCS12 microcontrollers

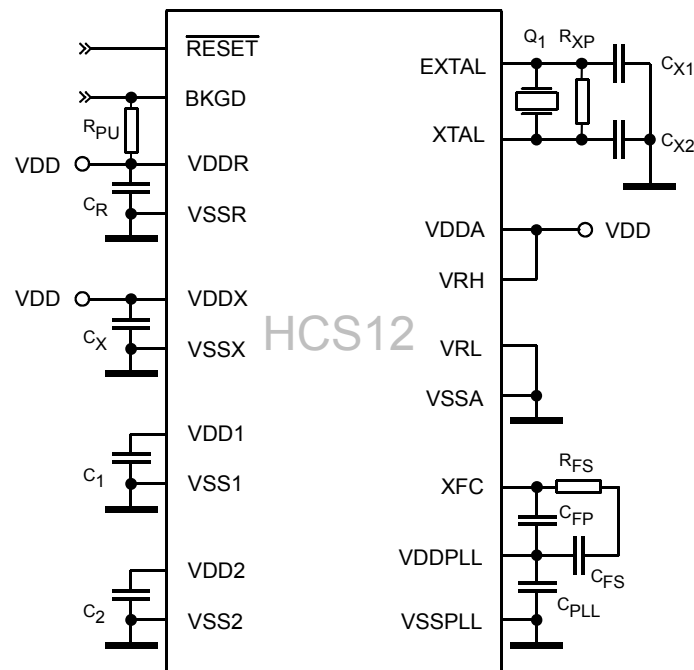


Introduction

The HCS12(X) are cost-effective rich-featured MCUs based on 16-bit HCS12(X) CPU. A wide range of derivatives can be composed of standard on-chip peripherals including:

- 2–32kB RAM,
- 0–4kB EEPROM,
- 32–512kB FLASH,
- *SCI (Special Communication Interface),*
- *SPI (Serial Peripheral Interface),*
- *IIC (Inter Integrated Circuit)* and additional modules like
- *TIM (Timer Interface Module),*
- *ADT (Analog To Digital Converter),*
- *CRG (Clock and Reset Generator Module),*
- *BDC (Background Debug Controller),*
- *EMAC (Ethernet Media Access Controller),*
- *PWM (Pulse Width Modulation module),*
- *USB (Universal Serial Bus),* etc.

In system programming (ISP) of HCS12(X) microcontrollers is performed via the single-wire BDM (Background Debug Mode). Below are the most important requirements and advices to get high reliability while working with our programmers. For further information look at the MCU specific datasheet.


 Figure 1. Typical HCS12 system connection ^{*1}

Used signals:

Pin name	Function	Signal level
VDDX, VDDA, VDDR, VREGEN	supply voltage	3.3–5.0 V ^{*2}
VSSX, VSSA, VSSR, VSS1, VSS2, VSSPLL	ground	0 V
VDD1, VDD2, VDDPLL	core supply pins for bypass capacitor connection	2.5 V ^{*3}
RESET\	reset input	H, L
MODA, MODB	mode selection input	PD ^{*4}
BKGD/MODC	background debug communication i/o, mode selection input	H, L, PU
XFC	pin for PLL circuitry filter	~ ^{*5}
EXTAL	oscillator, clock input	H, L
XTAL	oscillator output	H, L

^{*1)} For exact pin names and passive component values look at the derivative specific datasheet.

^{*2)} Depends on specific HCS12 derivative.

^{*3)} Generated internally (if voltage regulator is enabled)

^{*4)} Internal pull-down, active during reset. Some package types haven't these pins bonded out.

^{*5)} Slightly altering analog signal.

Table 1. HCS12 MCU ISP related signals description

Recommended target circuit design

In the following, you can find important notices applying to recommended connection of target MCU to the target system.

Our programmers BDM comply with Motorola/Freescale BDM connection specification. So, 6-pin header with 2 pins unconnected can be used in design stage of target system.

Purpose of the R1 resistor is to isolate the programmed chip from rest of target system. Recommended value of resistors for particular programmer is specified in *Device info* (see *Figure 5*). You can also use jumpers instead of the resistors.

Algorithm expects, that pins MODA, MODB (if available) are neither PULLED HIGH nor tied to VDD. During reset, BKGD is driven LOW, MODA, MODB is assumed to be internally PULLED DOWN, then the MCU enters **Special Single Chip Mode**. This allows programmer to communicate through BDM.

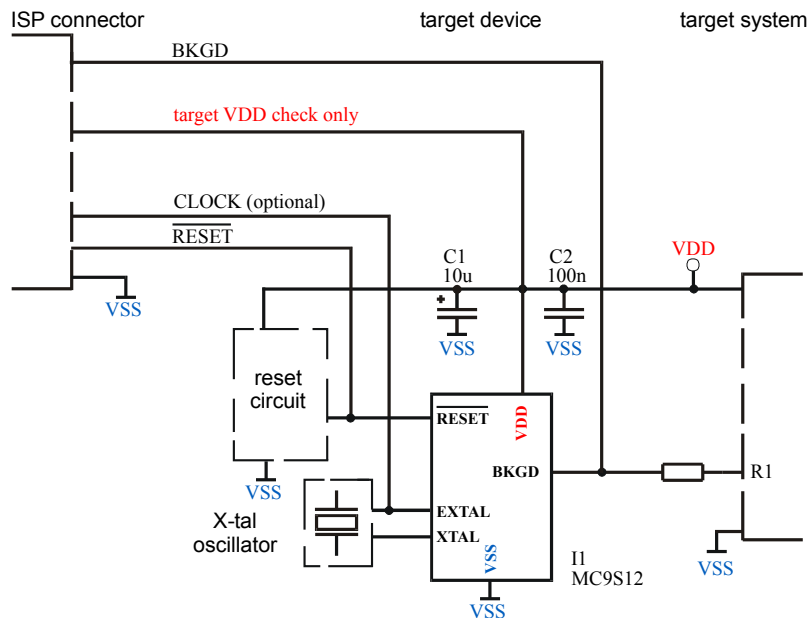


Figure 2. Recommended target circuit design using BDM interface

Note: Because communication with target board is performed only by 1 wire (pin BKGD), programming speed may vary depending on system configuration and programmer control SW settings (from x-sec to x-minutes per 32kB of FLASH). We recommend to engage MCU's PLL circuitry to get MCU running on appropriate bus/bdm frequency.

Because of high frequencies and fast signal transitions, please, do not omit the bypass capacitors as proposed by manufacturer. More information on this can be found in MCU's datasheet, section "PCB design recommendations" or "Recommended Printed Circuit Board Layout".

Device operation options

You can edit the Device operation options in menu *Operation options* of control program (Device->Device options->Operation options <Alt+O>).

In the case that your programmer supports target system power supply and you intend to use this option, it is needed to set supply voltage parameters according to your requirements.

You may also select level of ISP signals after operation (see *Figure 3*). Description of particular parameters you can find in the menu *Help* <F1>.

Figure 3. Settings ISP target supply <Alt+O>

There are some additional options to give you more adjustability. Besides *Target system power*, the programmer can provide an *optional clock* signal (2 different frequencies).

If you prefer to use your own system oscillator (or clock), there is another option, which allows you to get the best performance – engaging MCU's PLL. This option is available in 3 different frequencies (in case of active PLL, please be sure, you have connected the PLL filter between XFC and VDDPLL pins, otherwise the PLL frequency will be unstable and communication will fail).

Notice: may be, the programmer will not be able to communicate through BDM at your system frequency. In this case, the only way to solve this problem is to engage MCU's PLL. Programmer will set PLL to give the appropriate BDM frequency.

If you haven't provided any clock source to the MCU, it will run in self-clock mode at about 1MHz~5MHz. This can be fast enough, but only for small FLASH capacities ^{*6}.

Figure 4. Additional options <Alt+O>

^{*6}) Remark: Programming algorithm skips blank (\$FF) pages of FLASH and blank words of EEPROM areas to be programmed. Therefore, sometimes the programming procedure can take a shorter time as verification, and programming time may vary depending on programmed data.

Table 2. shows all possible MCU clock sources/modes/settings while operating with our programmers.

Oscillator	Optional clock	PLL	MCU runs on
no	off	off	self-clock
no	off	on	self-clock
no	on	off	optional clock
no	on	on	PLL clock derived from optional clock
yes	off	off	oscillator clock
yes	off	on	PLL clock derived from oscillator ^{*7}
yes	on	off	optional clock
yes	on	on	PLL clock derived from optional clock

**7) Recommended*

Table 2. HCS12 clock source settings/dependence

Device information

Additional information about selected MCU, such as ISP connector pin numbering, recommended target circuit design for particular MCU etc., you can find in the menu *Device info* of control program (Device->Device info <Ctrl+F1>) (see Figure 6).

Device info

Common information | ISP connection details | Part number description

Description of ISP connector pins:

- 1 - Target VDD check only
- 2 - Don't connect!
- 3 - RESET\
- 4 - Don't connect!
- 5 - Target System Supply Voltage *1
- 6 - BKGD
- 7 - VSS
- 8 - CLOCK (optional)
- 9 - VSS
- 10 - Don't connect!

Notes:

*1 Programmer can provide a power supply for the target system. See please Help for "Device options / Operation options" item for details.

Recommended target circuit design

ISP connector | target device | target system

The diagram shows an MC9S12 microcontroller connected to an ISP connector. The ISP connector pins are connected to the target device as follows: BKGD to BKGD, target VDD check only to VDD, CLOCK (optional) to XTAL, RESET to RESET, and VSS to VSS. The target device is connected to the target system VDD and VSS. A reset circuit and an X-tal oscillator are also shown connected to the target device. Capacitors C1 (10n) and C2 (100n) are connected between VDD and VSS. Resistor R1 is connected between VDD and VSS. The target device is labeled I1 MC9S12.

Device info

Common information | ISP connection details | Part number description

Manufacturer: Motoro.
Type: MC9S12
8-bit bytes: 3DC000
Organization: 3DC000
Algorithm name: Specia.

Supported by programme:
 BeeProg (Note: via I

ISP Note:
 The programmer is work: be made through ISP co The ZIF socket of the } forgotten in the ZIF s

General Info:
 The buffer data layout customize this option <Alt+S>.

Buffer data layout des:
 Default line:
 Addr:
 FLASH Page:
 3Ch 010000h - 013:
 3Dh 014000h - 017:
 3Eh 018000h - 01B:
 3Fh 01C000h - 01F:
 FLASH Protection byte:
 01FF0Dh

If needed, you can use the clock signal provided by progr clock can be adjusted in menu Device operation options <Alt+O>.


Purpose of the R1..R1 resistors is to isolate the programmed chip from rest of target system. Recommended value of resistors R1..R1 for BeeProg

OK


Figure 6. Device info


Good advices and troubleshooting


Connecting programmer to target system:

 **Turn off power supply of system** before connecting/ disconnecting programmer to/from system.


Before starting an operation:


 Before starting an operation with target MCU, please make sure, that the **ISP cable is correctly connected** to the target system and programmer. Also make sure that no device is inserted to *ZIF* socket of the programmer.

 **Details about pins assign for each MCU** and short description of circuit design you can find in control program (*Device Info <Ctrl+F1>*).


 Correctly selected values of resistor R1 provide reliable signal level recognition (for both, programmer and system) and ensure successful finishing of desired operation (see **Recommended target circuit design**).

If something goes wrong:

 If the programmer reports **signal interference error**, may be, a signal interference occurred between programmer and target system. Please make sure, your design meets connection recommendations. Check the minimal values of resistor R1, from programmer's point of view (in order to programmer be able to put L/H level on the pin).

 If the programmer reports that the **device does not respond**, or the **operation with MCU behaves suspiciously**, check the following:

1. MODA, MODB mode selection inputs are neither Pulled-Up nor tied to VDD. They are internally Pulled-Low to enable Normal or Special Single chip mode operation.
2. System frequency may result in BDM frequency, which don't comply with programmer's capability. If operations still give errors, you should engage MCU's PLL circuitry to achieve the appropriate BDM frequency.
3. RC network on RESET\ pin (if connected) hasn't long time constant.
4. If running on PLL, the PLL filter **must** be used (XFC, VDDPLL).

 Be aware, that longer ISP cable (longer than 20cm/0,7ft) may cause an unpredictable signal interference. Make sure you are using correct cable.

Revision history

07/2006:
Initial Release.