FIRE Emulator for HC12/MCS12

Version 18-Nov-2020

```
E::d.l
 addr/line  code              label      mnemonic                  comment
   P:00C886 E7EAD01D                     tst     -2FE3,Y
   P:00C88A 2720                         beq     0C8AC
                              {
        675                           primz = i + i + 3;
   P:00C88C 59                           asld
   P:00C88D C30003                       addd    #3
   P:00C890 6C84                         std     4,SP
        676                           k = i + primz;
   P:00C892 E380                         addd    0,SP
   P:00C894 2008                         bra     0C89E
                                      {
        679                                     flags[ k ] = FALSE;
   P:00C896 B746                         tfr     D,Y
   P:00C898 69EAD01D                     clr     -2FE3,Y
        680                                     k += primz;
   P:00C89C E384                         addd    4,SP
   P:00C89E 6C82                         std     2,SP
        677                           while ( k <= SIZE )
   P:00C8A0 8C0012                       cpd     #12
   P:00C8A3 2FF1                         ble     0C896
                                      }
        682                           anzahl++;
```

```
E::r
S   S  A       0   SP >00
X   X  B      0A   +01 0A
H   _  D      0A   +02 00
I   I  IX     13   +03 1B
N   _  IY     0A   +04 00
Z   _  SP  0DFE4   +05 13
V   _  PC  0C88C   +06 00
C   _  CCR   0D0   +07 07
```

```
E::v.w
flags = (1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0,
\\cos12\cos12\ast = (word = 0x0,
                     count = 12346,
                     left = 0x0D199,
                     right = 0x0,
                     field1 = 0,
                     field2 = 1)
```

For general information about the In-Circuit Debugger, refer to the **"FIRE User's Guide"** (fire_user.pdf). All general commands are described in **"PowerView Command Reference"** (ide_ref.pdf) and **"General Reference Guide"**.

Warning

NOTE:	Do not connect or remove probe from target while target power is ON. Power up: Switch on emulator first, then target Power down: Switch off target first, then emulator

Before debugging can be started, the emulator must be configured by hardware and software. At FIRE12 all the configuration can be done by software.

Ready to run setup files for most standard compilers can be found on the software CD in the directory **…/Demo/M68HC12/Compiler**. All setup files are designed to run the emulator stand alone without target hardware.

The following description gives assistance to create the initial setup (to run the emulator together with the target hardware of a specific application). It describes a typical setup with the most frequently used settings. It is recommended to use the programming language PRACTICE to create a batch file, which includes all necessary setup commands. PRACTICE files (*.cmm) can be created with the PRACTICE editor pedit (Command: **PEDIT <file name>**) or with any other text editor.

A basic setup file includes the following parts:

1. Select CPU and operation mode.
2. Set system options.
3. Select dualport mode (optional).
4. Set mapper.
5. Select frequency (optional).
6. Activate the emulator.
7. Initialize registers and chipselect units (optional).
8. Load the application file (optional).
9. Start application.

Now a typical example, how to setup the system:

1. Select CPU

   The system window controls the CPU specific setup. Please check this window very carefully and set the appropriate options. Before any setting is done, the SYStem window should be resetted to its default values with the command **SYStem.RESet**. This will switch the system down and guarantees a well defined position to start.

   Generally the emulator recognizes which derivative of the HC12 family is in use, but there are derivatives which differ only in a few on-chip peripherals. In this case the actual used CPU should be selected with the command **SYS.CPU <type>**.

2. Set **SYStem** Options.

3. Select **dualport mode** (optional).

   Dualport allows access to emulation RAM, while emulation is running. This is necessary to display variables, set breakpoints or display the flag listings while the emulation is running. **System.access** selects how dualport access is done and if it is allowed to be done. If there are no special requirements, SYStem.Access MIXed (default value) should be used for initial setup.

4. Set **mapper**.

   The mapper controls the memory access of the CPU. The Fire12 in Single Chip Mode needs no mapping commands. In Expanded Modes the Mapper is used to decide if target memory is used at a certain location of memory, or it emulation memory is used to replace the target memory.

```
map.reset                    ; reset mapper
map.ram 0x288000++3FFF       ; Map emulation Ram to PAGE 0x28
map.intern 0x288000++3FFF    ; Use emulation RAM for PAGE 0x28
```

5. Select **frequency** (optional)

   The CPU can be clocked by internal (emulator) or external (target). If the internal clock is used, the clock is provided by the VCO of the emulator. The setting of the internal clock is done by the **VCO** command.

```
VCO.Clock 40.                ; select frequency (necessary if
                             ; internal clock used)
                             ; set to 40 MHz -> the Bus works at
                             ; 20 MHz
```

6. **Activate** the emulator

   After activating the emulator with the `SYStem.Mode EmulInt` command you should be able to access memory (**Data.dump**, **Data.List**) and registers (**Register.view**).

```
SYStem.Mode EmulInt          ; emulation with target and internal
                             ; clock
SYStem.Mode AloneInt         ; emulation without target and internal
                             ; clock
```

7. Initialize **registers** and chipselect units (optional)

For correct loading of source files and correct display of code and symbolic information in **Data.List** and **Analyzer.List** it necessary to initialize chipselect unit and memory expansion.

```
Data.Set d:0x038 0x0F     ; specify which lines of Port G are used
                          ; as general purpose I/O and which are
                          ; used as address lines by writing MXAR
Data.Set d:0x03C 0x030    ; activate used chip selects by writing
                          ; CSCTL registers
Data.Set d:0x037 0x0C0    ; activate used windows by writing
                          ; WINDEF registers
```

8. **Load** application file

Application can be loaded by various file formats. The format depends on the compiler..

```
Data.Load.COSMIC file.cos12 /verify    ; load application file and
                                       ; verify if it is written
                                       ; correct to memory
```

9. **Start** application

Open the source file window with the command **Data.List**. This window shows the source code in the area next to the program counter. If the start address is not given by the compiler's output the program counter must be set with the command **Register.Set pc <startaddress>**. Now the application can be started with the command Go and stopped with Break or with the according buttons. If you want to run the application till a specific address is fetched the command Go <address> or Go <label> can be used. For example **"Go main"** starts the application and stops at label main.

Hang-up

If you are not able to stop the emulation, there may be some typically reasons:

Reset	The emulator can not deal with resets from Clock Monitor or watchdog (COP). Be sure to avoid this errors. This errors could be detected by searching the analyzer for the specific addresses (**Analyzer.List**, **Analyzer.Find** …).
Clock Error	The clock lines between the target and the CPU on the probe are very short. Therefore normally no problems should occur when using an external crystal. Be sure that the capacitors on the target have a value of 20 pF minimum and are with short routes connected to the CPU socket.

Dual-Port Errors

To solve problems with dualport error first increase the **SYStem.TimeReq** value. Be sure that the **SYStem.TimeOut** value is bigger than the access time limit. If it is not possible to solve the problem by changing the values, you must switch to **DENIED** mode. In this mode no hidden access to emulation memory is possible while running real-time emulation. The dual-port access has no effect on CPU performance.

Debugging via VPN	**The debugger is accessed via internet/VPN and the performance is very slow. What can be done to improve debug performance?** The main cause for bad debug performance via internet or VPN are low data throughput and high latency. The ways to improve performance by the debugger are limited: • in practice scripts, use "SCREEN.OFF" at the beginning of the script and "SCREEN.ON" at the end. "SCREEN.OFF" will turn off screen updates. Please note that if your program stops (e.g. on error) without executing "SCREEN.OFF", some windows will not be updated. • "SYStem.POLLING SLOW" will set a lower frequency for target state checks (e.g. power, reset, jtag state). It will take longer for the debugger to recognize that the core stopped on a breakpoint. • "SETUP.URATE 1.s" will set the default update frequency of Data.List/Data.Dump/Variable windows to 1 second (the slowest possible setting). • prevent unneeded memory accesses using "MAP.UPDATEONCE [address-range]" for RAM and "MAP.CONST [address--range]" for ROM/FLASH. Address ranged with "MAP.UPDATEONCE" will read the specified address range only once after the core stopped at a breakpoint or manual break. "MAP.CONST" will read the specified address range only once per SYStem.Mode command (e.g. SYStem.UP).

The FIRE12 emulation head supports all 68HC12 derivatives from Freescale Semiconductor. The adaptation to different probes is done by changing the module.The maximum frequency of the base module is 50 MHz (25 MHz ECLK) for Single Chip applications and 40 MHz for expanded applications, however the emulation is only possible to the max. speed of the MCU available from the chip manufacturer. All emulation probes support only single chip mode. The probes can be used at any voltage

The emulator can be equipped with an additional modul called port analyzer to get timing and state trace features for all MCU I/O ports.

SYStem.Mode — Emulation modes

Format:	**SYStem.Mode** <*mode*>
<*mode*>:	**ResetDown**
	AloneInt
	AloneExt
	EmulInt
	EmulExt

The emulation head can stay in 6 modes. The modes are selected by the **SYStem.Up** or the **SYStem.Mode** command.

Reset	Target is down, all drivers to the target are in tristate mode.
Alone Internal	Probe is running with internal clock, drivers inactive.
Alone External	Probe is running with external clock, drivers inactive.
Emulation Internal	Probe is running with internal clock, strobes to target are generated.
Emulation External	Probe is running with external clock, strobes to target are activated.

In active mode, the power of the target is sensed and by switching down the target the emulator changes to **RESET** mode. The probe is not supplied by the target. When running without target, the target voltage is simulated by an internal pull-up resistor.

SYStem.Access — Define method for dual-port access

Format:	**SYStem.Access** <*mode*>
<*mode*>:	**FreeCycle** **CPU** **MIXed** **Denied**

FreeCycle The dual-port access is made on unused CPU accesses.

CPU The dual-port access is made by the Background Debug Interface of the CPU. Only the actual 64K memory map is visible. So paged areas may be shown incorrect.

MIXed This dual-port access is a mixture of the Modes FreeCycle and CPU.

Denied No dual-port access is allowed while the real-time emulation is running. This mode must be used if the CPU is hold in stop state for long time or if the clock signal is switched off.

In the mode **FreeCycle** the dualport access enables to read or write emulation memory while the CPU is executing code.

In the mode **CPU** the dualport access is done by the CPU. The memory which is accessible in this way is the memory which is reachable for the CPU at the actual point of time. So read and write operations can be done to internal RAM and registers and even to target memory, while the CPU executes code.

The built in mechanism for hidden accesses does not to take care of paged areas. It will use the actual page pointers which may change during real time execution. So the contents of a window may change while viewing a paged ram location

Pay attention to the mapping precedence of the CPU' s internal resources (compare chapter “Resource mapping” in the CPU Technical Summary”) and the mapping on the emulator (**MAP.List**)!

The mode **MIXed** combines the modes FreeCycle and CPU. All unpaged areas are accessed by CPU, while the paged areas are accessed with FreeCycle. So for a Single Chip Application MIXed is the most convenient selection.

So there might be a difference in the accessible RAM dependent from the access mode. For example if the CPU's internal ram is mapped to 02000h and SYStem. Access is switched to CPU, then **Data.dump e:02000** shows the contents of the CPU's internal ram while the same window shows the contents of the emulation memory if SYStem.Access is switched to FreeCycle.

SYStem.CPU — CPU select

Format:	**SYStem.CPU** <*type*>
<*type*>:	**M68HC12B | M68HC12BC | M68HC12D | M68HC12DA | M68HC12DG | M68HC12DT | MCS12DP**

With this command the processor type is selected. The emulator recognizes which probe is in use and selects the CPU type depending on this information. Some probes support more than one derivative of HC12 family. In this case the actual used CPU should be selected with the command SYS.CPU <type>.

The selected CPU can be emulated in **Single Chip Mode**, To provide the single chip modes the emulator uses a Port Replacement Unit.

Restrictions

MODE-Register	Don’t change this register!
PEAR-Register	Don’t change this register!
No Flags for misaligned word accesses	Internal memory can be read or written with word accesses to odd addresses. The Flag memory mechanism can not deal with these cycles, so no flag is set in this case.
No C1 Flag at Single Step	The C1 Flag shows if a branch is taken. This does not work if the according command is executed by a single step when the emulator is configured to work in assembler mode.
No Program Break in ROM areas	Program and HLL breakpoints are not executed if set to Read Only Memory areas.
On-Board Programming	On-Board Programming with external programmers is not allowed with the emulation probe (may be damaged).
Port Replacement HC12B, HC12D, MCS12DP	The ports A, B, E and K are replaced by a Port Replacement Unit. This unit does not support the pull up resistors and reduced drive modes.

SYStem.LOCK

Multicore applications

Format:	**SYStem.LOCK [ON | OFF]**

SYStem.LOCK is needed on multicore applications. This feature is not supported on HC/HCS12 applications.

Format: **SYStem.Option** <*option*>

<*option*>: **TraceAll [ON | OFF]**
TraceWait [ON | OFF]
TraceRes [ON | OFF]

TraceAll (currently not available)
Normally dual port cycles and cycles of the CPU´s firmware are not traced by the analyzer. All CPU cycles are traced if this option is set ON.

TraceWait (currently not available)
Normally wait cycles. The cycles between the WAIT command and the next opcode fetch cycle.

TraceRes (currently not available)
The reset line on the CPU disables the trace analyzer. To see also this cycles switch on this option.

SYStem.Option PerRest Reset target

Format: (currently not available)
SYStem.Option PerReset [ON | OFF]

When activated, the reset output line is active while the system is down. This ensures that the target peripherals are in reset state after the emulation is activated.

SYStem.Option VDDPLL Enable PLL

Format: **SYStem.Option VDDPLL [ON | OFF]**

OFF VDDPLL is connected to GND (Only HC12)

ON VDDPLL is connected to VDD (Only HC12)

Enables the PLL on HC12 derivatives. On HCS12 derivatives the PLL is supplied all the time.

HCS12 Applications

Activating the COP without any special step will force the emulator's CPU to reset after the emulator entered break mode, because the COP is not triggered.

To solve this problem set the RSBCK (RTI and COP stop in Active BDM mode) in the COPCTL (COP Control) register. The COPCTL bit follows only the first write access. This can be used to disable the COP (in break mode) though the original application needs a different setting:

Add the command Data.Set 0x3C 0x40 to your Batchfile (*.cmm start file). Place it after the Data.Load command. If now the application attempts to clear RSBCK the bit will remain on high and so the application can be stopped without danger of getting a reset from the COP.

68HC12 Applications

Activating the COP without any special step will force the emulator's CPU to reset after the emulator entered break mode, because the COP is not triggered.

To solve this problem set the RSBCK (RTI and COP stop in Active BDM mode) in the RTICTL (Real-Time Interrupt Control) register. The COPCTL bit follows any write access so the application has to be modified if the default setting of RSBCK is zero.

Some HCS12 derivatives carry a debug block within their core which offers some trigger features and a small trace. Generally this unit is not needed, because its functionality is a subset to the emulator's features. In some special cases however an additional independent analyzer system may be helpful.

Onchip.Mode — Storage behavior

Format:	**Onchip.Mode** <*mode*>
<*mode*>:	**FIFO** **STACK**

Defines the storage behavior of the on-chip trace.

FIFO Storage occurs till a trigger stops the trace. If the trace buffer is full and new data is written, the least recent data will fall out of the trace.

STACK Storage starts with a trigger and stops when the trace buffer is filled or when the CPU is stopped.

Format:	**Onchip.Mode** <*mode*>
<*mode*>:	**FlowTrace** **LoopTrace** **DetailTrace** **EventTrace**

Defines the behavior of the on-chip trace.

FlowTrace	Trace records program flow.
LoopTrace	FlowTrace ignoring tight loops. See Core User Guide HCS12.
DetailTrace	All cycles except for fetch and free cycles are recorded.
EventTrace	Trace records are controlled by the on-chip trigger (Event modes).

Onchip.view Open control panel to configure on-chip trace

Format:	**Onchip.view**

Opens a control panel to configure the on-chip trace.

TrOnchip.CONVert — Adjust range breakpoint in on-chip resource

Format: **TrOnchip.CONVert [ON | OFF]**

The hardware breakpoints of the HC/HCS12 can only cover specific aligned ranges. When enabled (default) the on-chip breakpoints are automatically converted from a range to a single address if required. If the switch is off, the system will only accept breakpoints which exactly fit to the on-chip breakpoint hardware.

TrOnchip.VarCONVert — Adjust complex breakpoint in on-chip resource

Format: **TrOnchip.VarCONVert [ON | OFF]**

The hardware breakpoints of the HC/HCS12 can only cover specific ranges. If you want to set a marker or breakpoint to a complex variable, the on-chip break resources of the 9S08 CPU may be not powerful enough to cover the whole structure. If the option **TrOnchip.VarCONVert is on** the breakpoint will automatically be converted into a single address breakpoint. This is the default setting. Otherwise an error message is generated.

Format: **TrOnchip.Mode** <*mode*>

<*mode*>: **OFF**
BreakAORB
BreakATHENB
TraceAORB
TraceATHENB

Defines the way how matches of the comparators A and B are used together.

OFF	No Action.
BreakAORB	Break program if comparator A or comparator B match.
BreakATHENB	Break program if first comparator A and then comparator B match.
TraceAORB	Break on-chip trace if comparator A or comparator B match.
TraceATHENB	Break on-chip trace if first comparator A and then comparator B match.

TrOnchip.RESet Set to default values

Format: **TrOnchip.RESet**

Resets the on-chip trigger system to the default state.

TrOnchip.view Show control panel

Format: **TrOnchip.view**

Opens a control panel to configure the on-chip breakpoint registers.

Trigger.Set

Format:	**Trigger.Set BUSA [ON | OFF]**

Default: OFF.

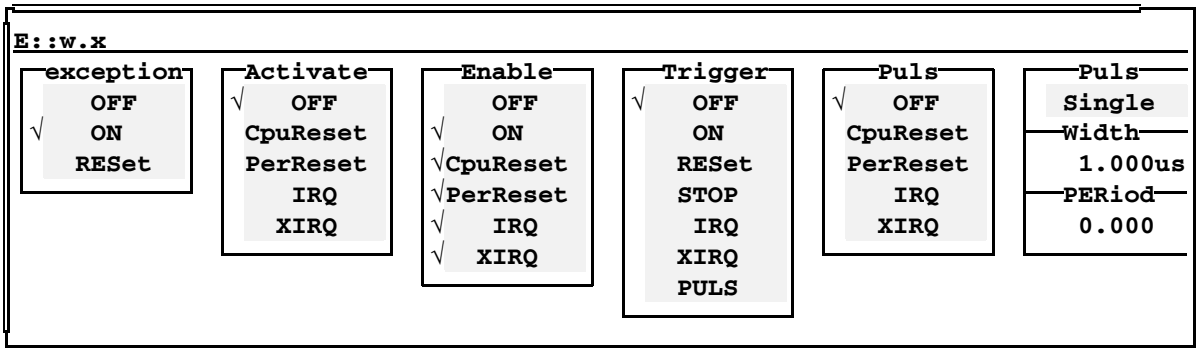
Enables the external trigger input. The program execution halts on a rising edge on the external trigger input.Trigger.Out.

Format:	**Trigger.Out BUSA [ON | OFF]**

Default: ON.

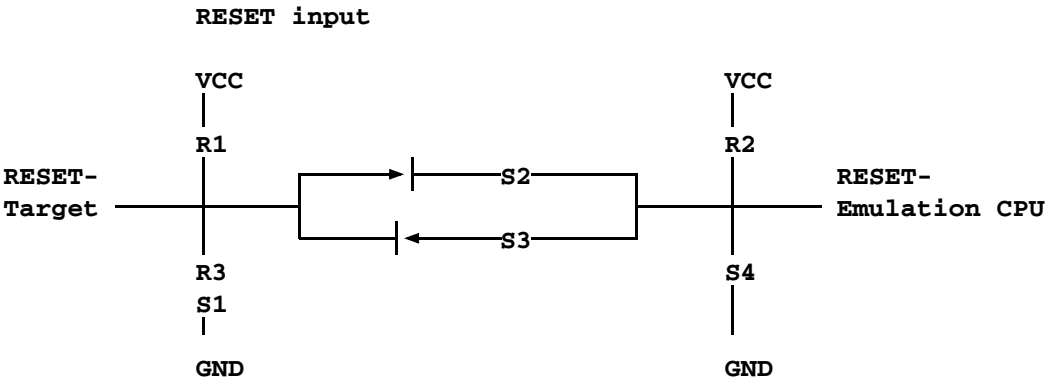
When enabled a high pulse of about 150 ns is asserted on the external trigger line when the user program execution halts.

Exception Control



RESET Control

The reset line (input and output) is controlled by a bridge with analog switches and diodes.




```
R1 = 220K
R2 = 4.7K
R3 = 22

S1   Reset Target    X.Activate PerReset
                     X.Puls PerReset * running

S2   Reset Out       running

S3   Reset In        X.Enable Reset * running

S4   Internal Reset  Emulator Control
                     X.Activate CpuReset * running
                     X.Puls     CpuReset * running
```

eXception.Activate

Force exception

Format:	**eXception.Activate RESET[ON | OFF]**
Format:	**eXception.Activate IRQ [ON | OFF]**
Format:	**eXception.Activate XIRQ [ON | OFF]**
Format:	**eXception.Activate OFF**

RESET Activates the RESET line.

IRQ Activates the IRQ line.

XIRQ Activates the XIRQ line.

OFF No activation of any exception line.

Format:	**eXception.Enable RESetOut [ON | OFF]**
Format:	**eXception.Enable RESetIm [ON | OFF]**
Format:	**eXception.Enabke IRQ [ON | OFF]**
Format:	**eXception.Enabke XIRQ [ON | OFF]**
Format:	**eXception.Enable OFF**
Format:	**eXception.Enable ON**

RESetOut Enables the RESET connection from CPU to target. If the CPU in the emulator goes to RESET, the RESET line on the target side gets active.

RESetIn Enables the RESET connection from target to CPU. If the Reset line on the target side gets active the CPU in the emulator will be driven to RESET.

IRQ Enables the IRQ line. IRQ events on the target will be transmitted do PortE1 of the CPU in the emulator.

XIRQ Enables the XIRQ line. XIRQ events on the target will be transmitted do PortE1 of the CPU in the emulator.

OFF Disable all exception line.

ON Enables all exception lines.

Format:	**eXception.Pulse RESET [ON | OFF]**
Format:	**eXception.Pulse IRQ [ON | OFF]**
Format:	**eXception.Pulse XIRQ [ON | OFF]**
Format:	**eXception.Pulse OFF**

RESET	Stimulate RESET line of the CPU.
IRQ	Stimulate on IRQ line.
XIRQ	Stimulate on XIRQ line.
OFF	No stimulation on any exception line.

EEPROM Management

The emulator supports easy writing to the internal EEPROM. It executes the necessary EEPROM program sequence instead of the normal write operation if a write access to the EEPROM is indicated. There are two ways to do this:

Using the storage class EEPROM

Any write access done by a command using the storage class EEPROM (shortening: EE) starts the EEPROM program sequence. So the EEPROM can be initialized with standard set or load commands using the storage class **EEPROM or EE:**

```
d.s EEPROM:0x0D00 2 3 4 5              ; setting bytes
d.s     EE:0x0D00 2 3 4 5              ; short form
d.s EEPROM:0x0D00++0x2ff 0x0ff         ; clear EEPROM
d.load.b eepromdat.bin  EEPROM:0D00    ; loading a binary file
                                       ; named eepromdat.bin
```

It is possible to load the application using the command

Data.Load <file> EEPROM:

because write accesses to RAM done by this command will not cause an error. The only disadvantage of this method is, that it takes a lot of time.

Description	
C:, P:, D:	Specify the same address-area (CPU-access)
A:	Absolute memory access (requires MMU-table)
EEPROM:	EEPROM write
E:	Emulation memory access (dual-ported)
AP:	Physical address (68HC12A4/F8/DA128/DG128 only)

C:, P: and D:

This storage classes operate on the same physically memory. They are only used to be compatible with other emulation probes. CPU internal registers and memory may not be accessed dual-ported, by mapping memory to the same address range data written to the internal memory are also present in the emulation memory.

EEPROM:

This storage class is used to program the internal EEPROM. On read cycles there is no difference to the access mode with **C:** or **D:**. On write cycles the monitor program executes an EEPROM write protocol.

```
Data.Set  EEPROM:0x0E00 0x12 0x34
D.s       EE:    0x0E00 0x12 0x34     EE: can be used as short form
```

SYStem.Option ROMHM — ROM in second half of map

Format: **SYStem.Option ROMHM [ON | OFF]**

The SYStem.Option ROMHM must be set if the bit ROMHM is set in the CPU's MISC register. In this case page 6 of the FLASH EEPROM is visible from $4000--$7fff.

Keywords for the Display

EAddress	Execution address from pipeline reconstruction
WR	Write cycle
DBE	Data Bus Enable
IRQ, XIRQ	Interrupt Inputs
LBSTR	Lower Byte Strobe
IPIPE0, IPIPE1	Pipeline Execution Status
DMOVE0, DMOVE1	Data Movement Status
RAWDUMMY	Read After Write Dummy Cycle

General 68HC12 Keywords for the Trigger Unit

Input Event	Meaning	Analyzer Hardware FEC
BYTE0	Access low byte	X
BYTE1	Access high byte	X
BYTE	Byte transfer	X
DATA	Data access	X
DUMMYCYCLE	CPU dummy cycle	X
HSTART	Start command from high byte	X
IRQ	Interrupt request	X
LSTART	Start command from low byte	X
OPFETCH	1st cycle of command	X
PIPEMOVED	CPU pipeline shifted in the next instruction	X
Read	Read access	X
RESET	Reset signal	X
WORD 2)	Word transfer	X
Write	Write access	X
XIRQ	XIRQ input	X

1) Information on Execution Starts and Data Movement in the pipeline are valid in the cycle after the cycle they refer to. So for usage in the analyzer trigger you have to write a multi level program.

Port Signals HC12DX128

Name	Group	Description
PA0 .. PA7	PORTA	PA PA0 .. PA7
PB0 .. PB7	PORTB	PB PB0 .. PB7
PE0 .. PE7	PORTE	PE PE0 .. PE7
PH0 .. PH7	PORTH	PH PH0 .. PH7
PCAN0 .. PCAN7	PORTJ	PJ PCAN0 .. PCAN7
PK0 .. PK3	PORTK	PK PK0 .. PK3
PK7	PORTK	PK PK7
PP0 .. PP3	PORTP	PP PP0 .. PP3
PS0 .. PS7	PORTS	PS PS0 .. PS7
PT0 .. PT7	PORTT	PT PT0 .. PT7

Port Signals MCS12DP

Name	Group	Description
PA0 .. PA7	PORTA	PA PA0 .. PA7
PB0 .. PB7	PORTB	PB PB0 .. PB7
PE0 .. PE7	PORTE	PE PE0 .. PE7
PH0 .. PH7	PORTH	PH PH0 .. PH7
PJ0 .. PJ1	PORTJ	PJ PJ0 .. PJ1
PJ6 .. PJ7	PORTJ	PJ PJ6 .. PJ7
PK0 .. PK7	PORTK	PK PK0 .. PK7
PM0 .. PM7	PORTM	PM PM0 .. PM7
PP0 .. PP7	PORTP	PP PP0 .. PP7
PS0 .. PS7	PORTS	PS PS0 .. PS7
PT0 .. PT7	PORTT	PT PT0 .. PT7

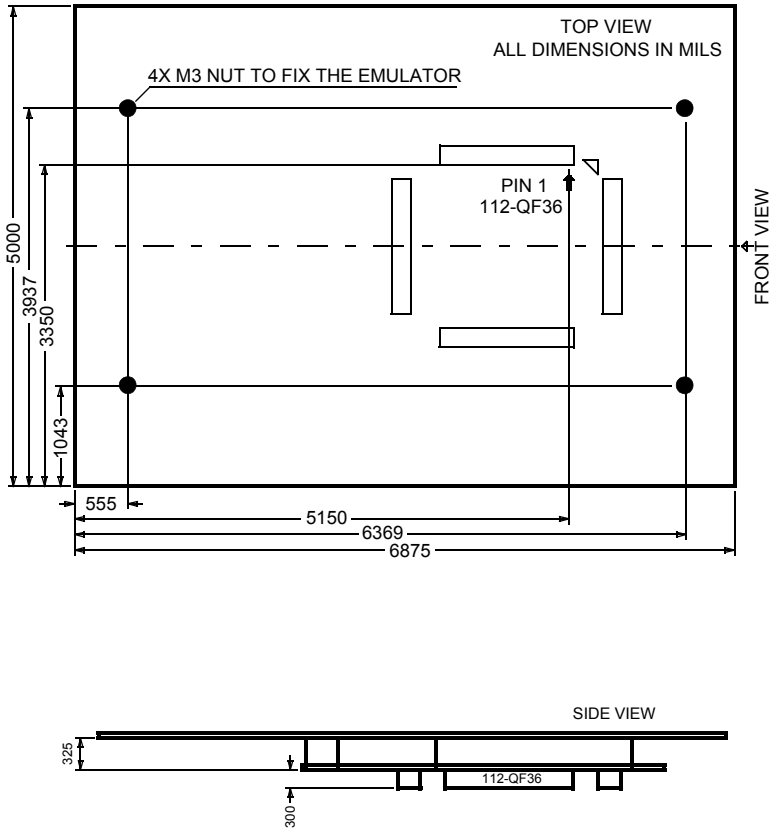
Port Signals MCS12X

Name	Group	Description
PA0 .. PA7	PORTA	PA PA0 .. PA7
PB0 .. PB7	PORTB	PB PB0 .. PB7
PC0 .. PC7	PORTC	PC PC0 .. PC7
PD0 .. PD7	PORTD	PD PD0 .. PD7
PE0 .. PE7	PORTE	PE PE0 .. PE7
PH0 .. PH7	PORTH	PH PH0 .. PH7
PJ0 .. PJ2	PORTJ	PJ PJ0 .. PJ2
PJ4 .. PJ7	PORTJ	PJ PJ4 .. PJ7
PK0 .. PK7	PORTK	PK PK0 .. PK7
PM0 .. PM7	PORTM	PM PM0 .. PM7
PP0 .. PP7	PORTP	PP PP0 .. PP7
PS0 .. PS7	PORTS	PS PS0 .. PS7
PT0 .. PT7	PORTT	PT PT0 .. PT7

Mechanical Dimensions

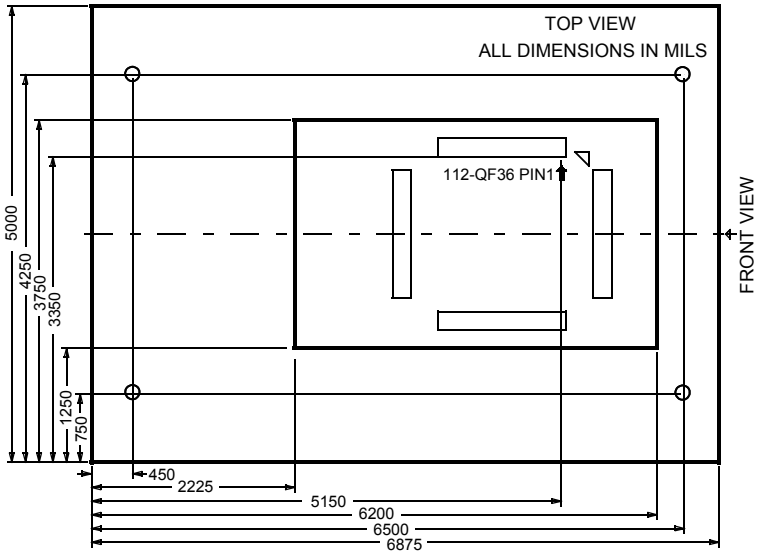
Dimension

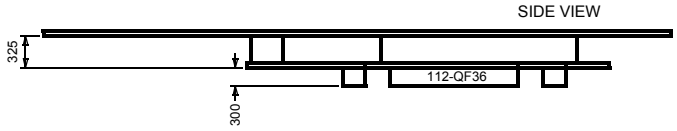
LA-9656 M-MCS12-B32



Dimension

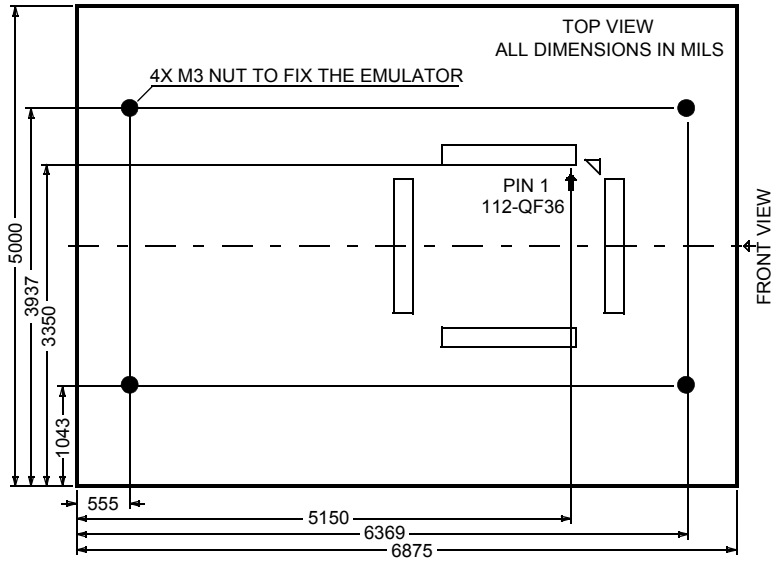
LA-9657 M-MCS12-DB128

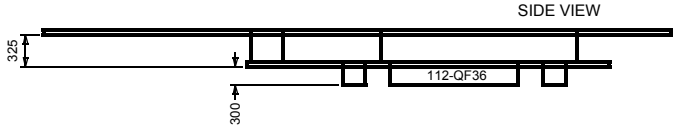




Dimension

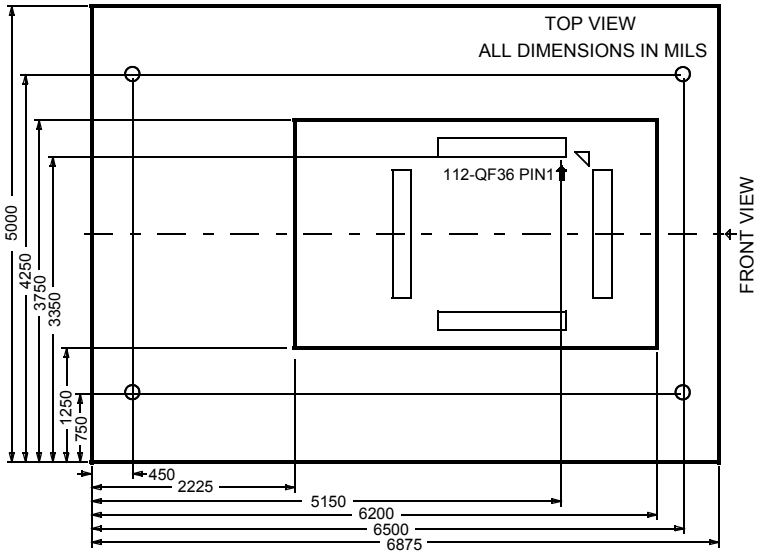
LA-9658 M-MCS12C

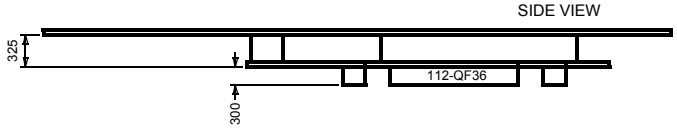




Dimension

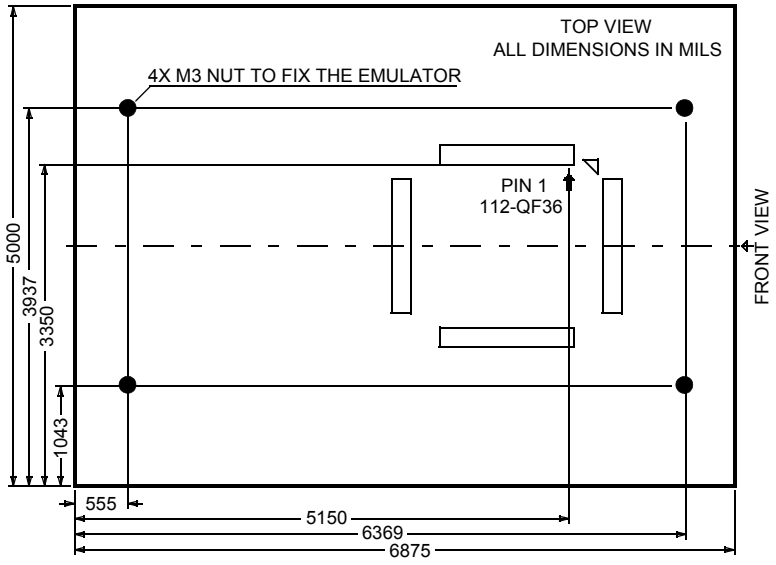
LA-9646 M-MCS12-DP256

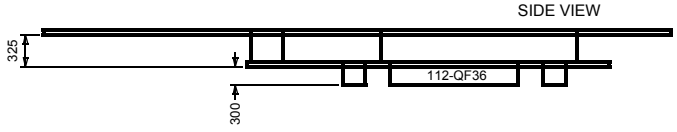




Dimension

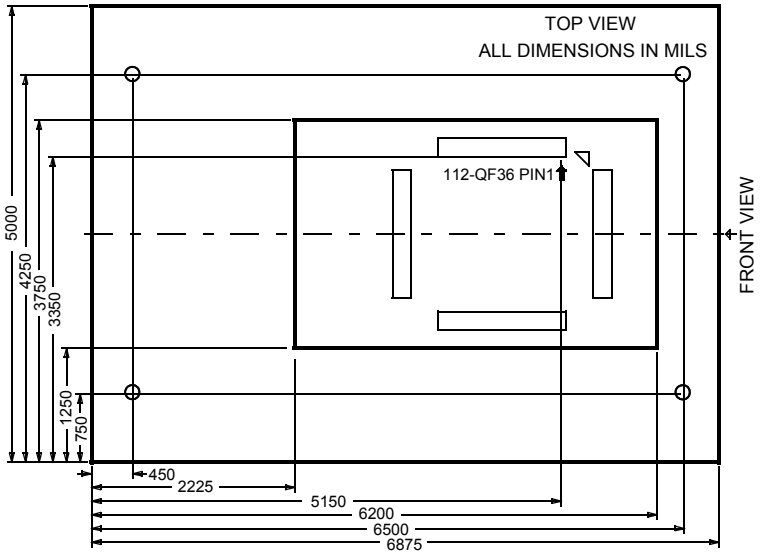
LA-9647 M-MCS12-D60

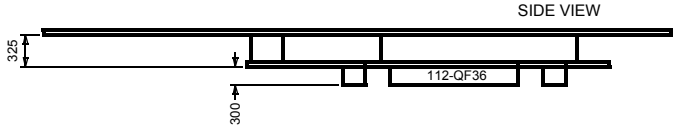




Dimension

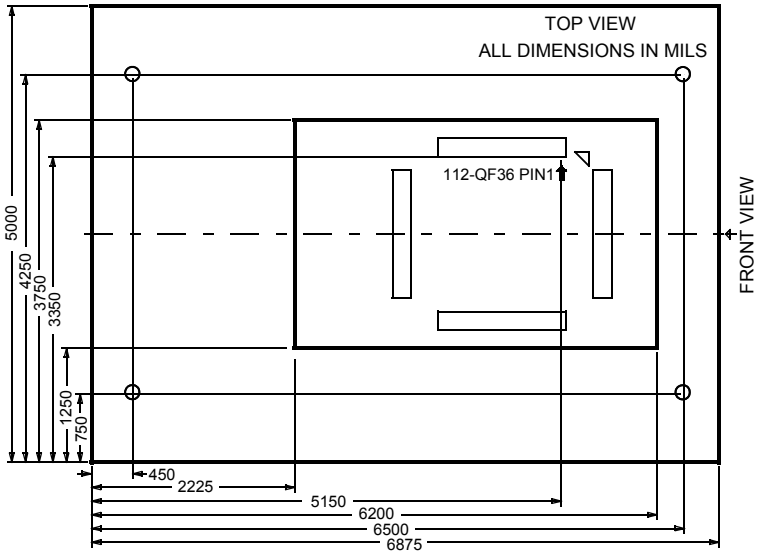
LA-9645 M-HC12-DG128

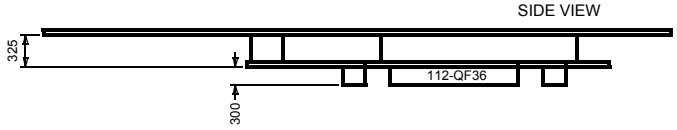




Dimension

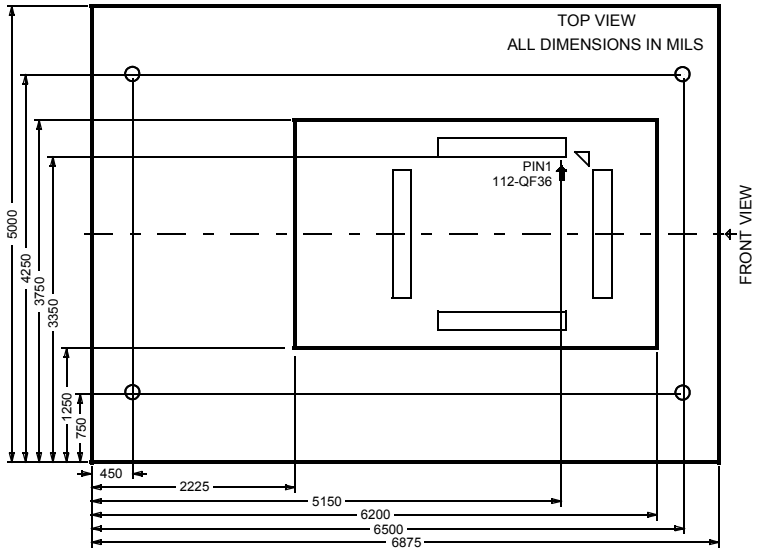
LA-9644 M-MCS12-DT128

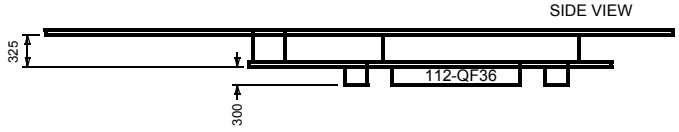




Dimension

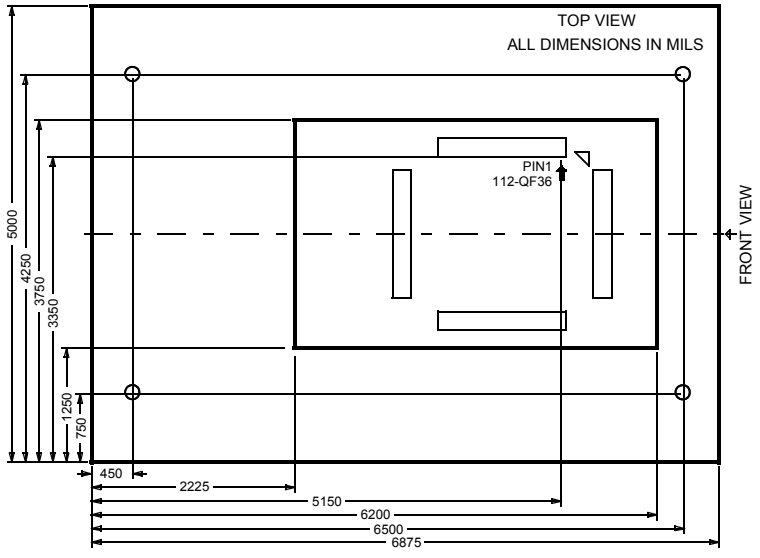
LA-9642 FIRE-MCS12-1MB

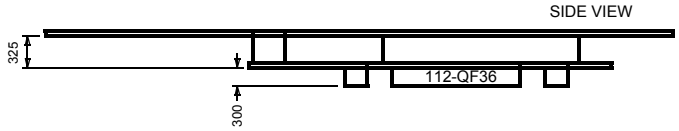




Dimension

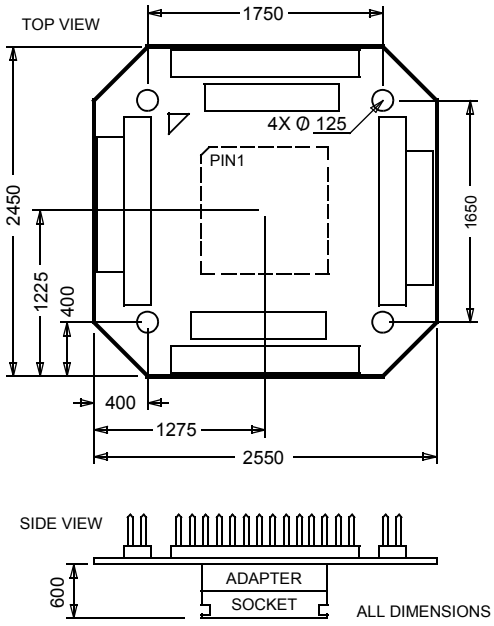
LA-9641 FIRE-MCS12-256K





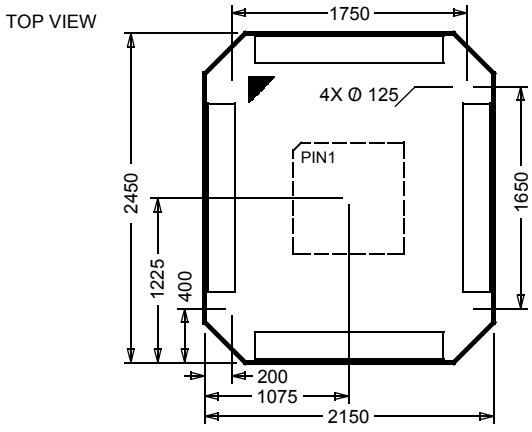
Dimension

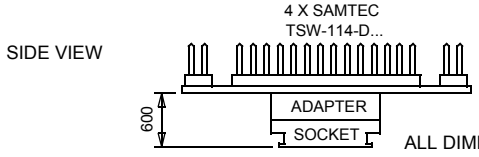
LA-9681 ET80-ETO-QF14/MSC12



ALL DIMENSIONS IN 1/1000 INCH

LA-9683 ET112-TO-TET52SB

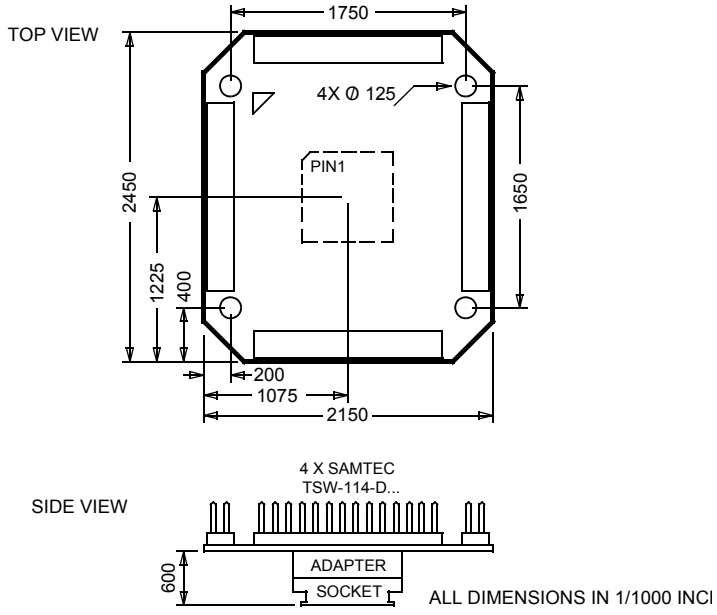




ALL DIMENSIONS IN 1/1000 INC

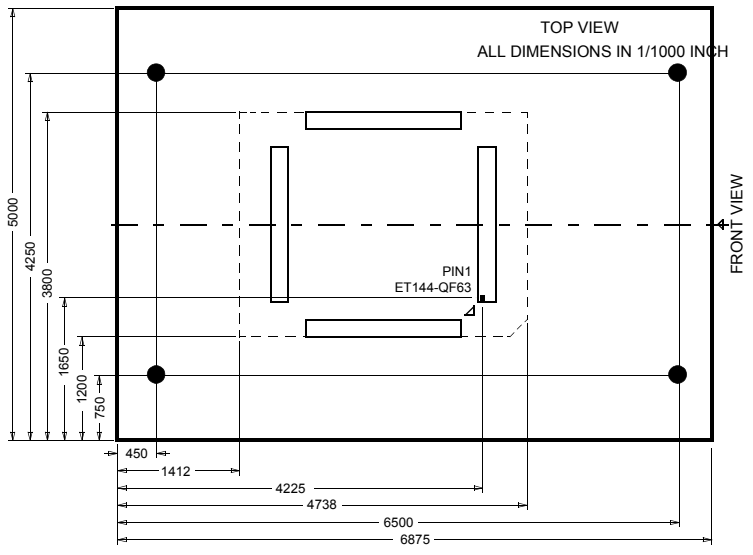
Dimension

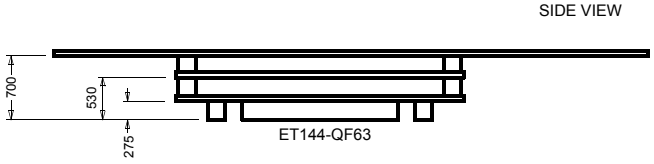
LA-9684 ET112-TO-TET48SD



Dimension

LA-9686 M-S12X-DP512



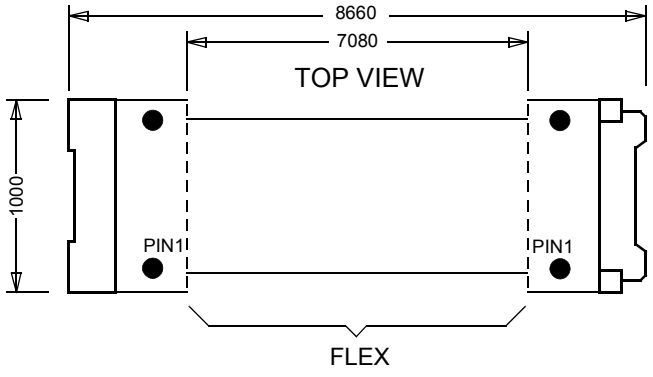


Dimension

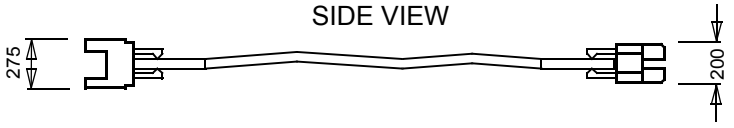
LA-9687 S12X-FLEXEXT-L

MICTOR-EXTENDER

HORIZONTAL-HORIZONTAL

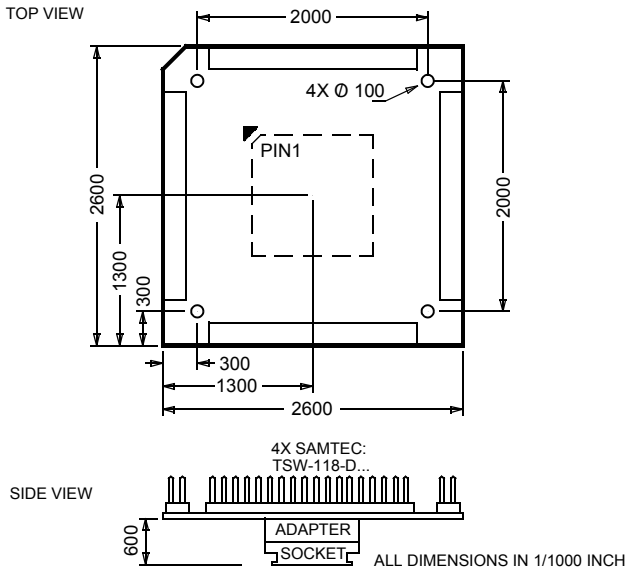


ALL DIMENSIONS IN 1/1000 INCH

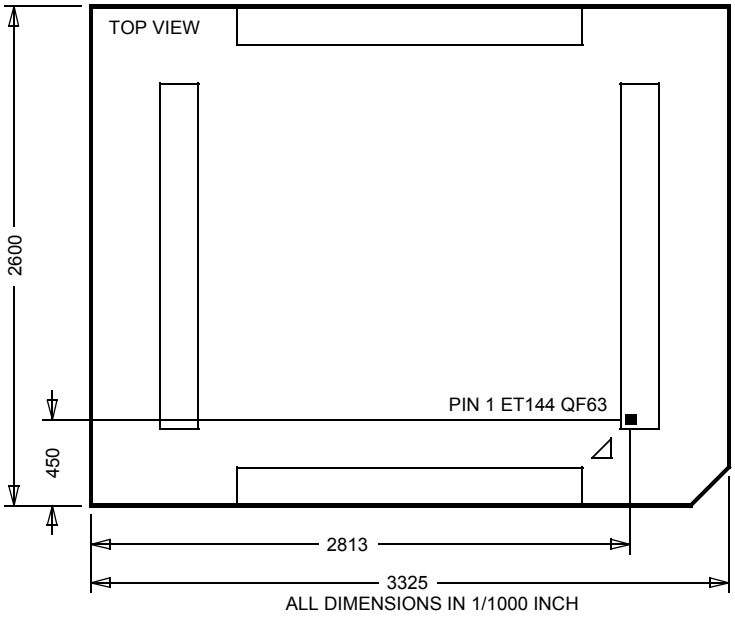


Dimension

LA-9688 ET112-ETO-QF36/S12X

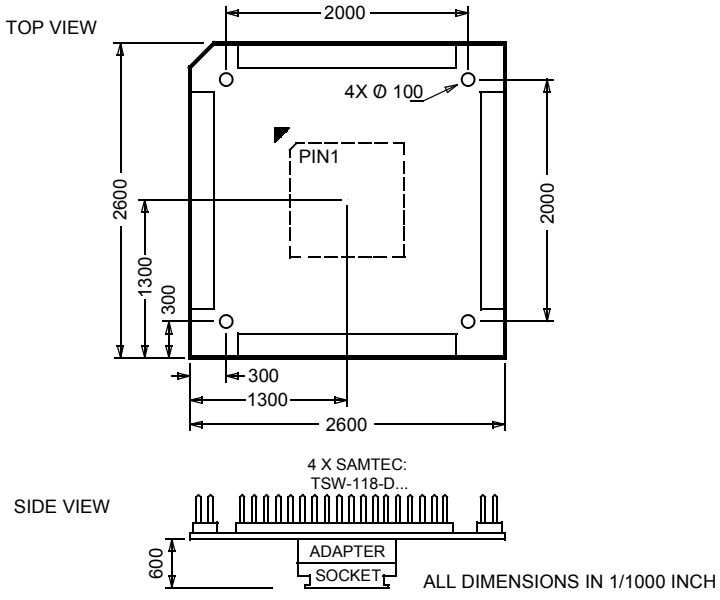


LA-9689 M-S12X-EP100

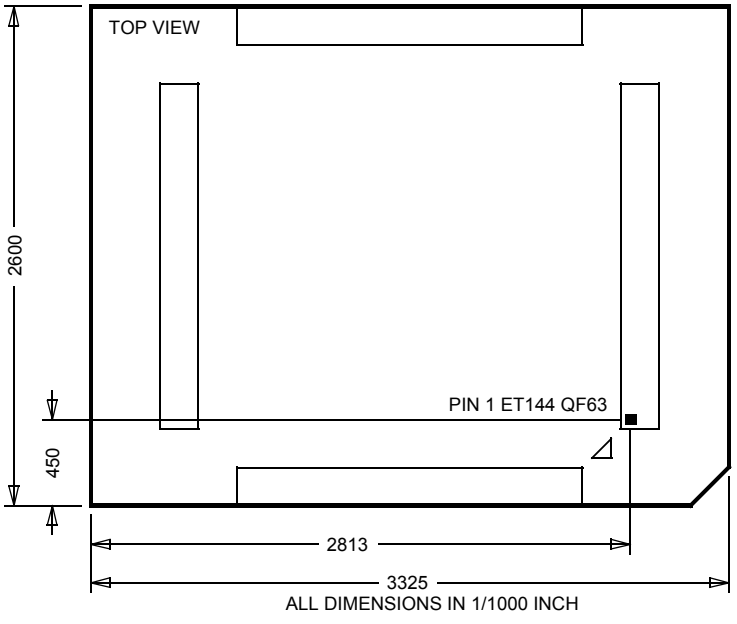


Dimension

LA-9690 ET80-ETO-QF14/S12X

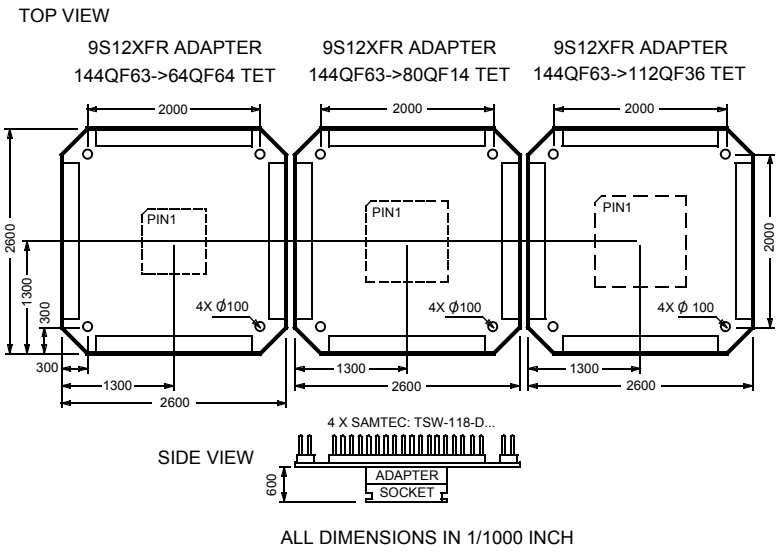


LA-9691 M-S12X-F512



Dimension

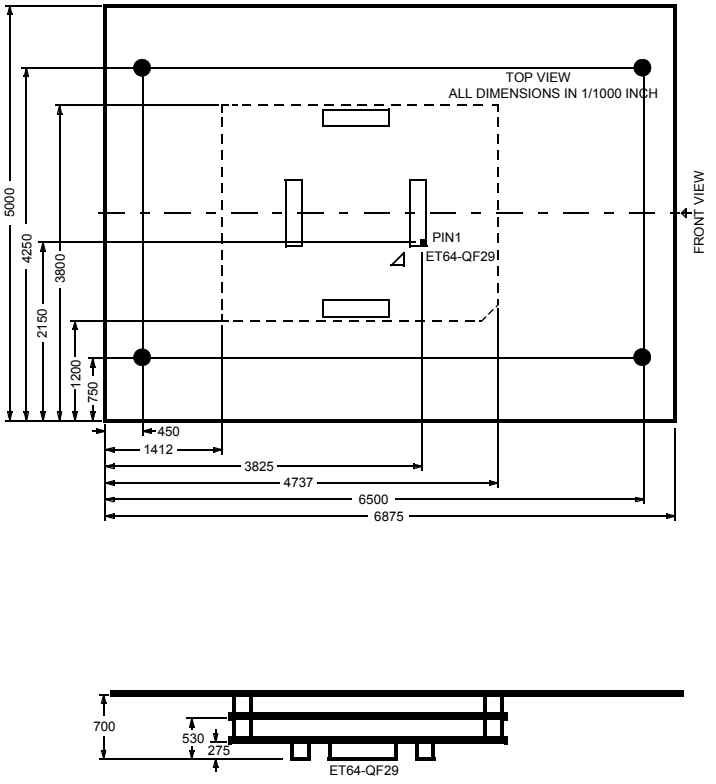
LA-9692 ET112-ETO-QF36/S12XF



ALL DIMENSIONS IN 1/1000 INCH

Dimension

LA-9704 M-HC08-GZ60



Adaptions

No adaptions available !

Adapters

No adapters necessary !

Operation Voltage

This list contains information on probes available for other voltage ranges. Probes not noted here supply an operation voltage range of 4.5 … 5.5 V.

CPU	Module	Adapter	Voltage Range
MC68HC912D60	LA-9647	-	2.7 .. 5.0 V
MC68HC912DG128	LA-9645	-	2.7 .. 5.0 V
MC68HC912DT128	LA-9644	-	2.7 .. 5.0 V
MC9S12A128	LA-9646	-	4.7 .. 5.3 V
MC9S12A128	LA-9646	LA-9680	4.7 .. 3.5 V
MC9S12A256	LA-9646	-	4.7 .. 5.3 V
MC9S12A256	LA-9646	LA-9680	4.7 .. 3.5 V
MC9S12A64	LA-9646	LA-9680	4.7 .. 3.5 V
MC9S12B128	LA-9646	-	4.7 .. 5.3 V
MC9S12B128	LA-9646	LA-9680	4.7 .. 3.5 V
MC9S12B256	LA-9646	-	4.7 .. 5.3 V
MC9S12B256	LA-9646	LA-9680	4.7 .. 3.5 V
MC9S12B64	LA-9646	-	4.7 .. 5.3 V
MC9S12B64	LA-9646	LA-9680	4.7 .. 3.5 V
MC9S12C128	LA-9658	-	4.7 .. 5.3 V
MC9S12C32	LA-9658	-	4.7 .. 5.3 V
MC9S12C64	LA-9658	-	4.7 .. 5.3 V
MC9S12C96	LA-9658	-	4.7 .. 5.3 V
MC9S12D64	LA-9646	-	4.7 .. 5.3 V
MC9S12D64	LA-9646	LA-9680	4.7 .. 3.5 V
MC9S12DB128	LA-9657	-	4.7 .. 5.3 V
MC9S12DG128	LA-9646	-	4.7 .. 5.3 V
MC9S12DG128	LA-9646	LA-9680	4.7 .. 3.5 V
MC9S12DG256	LA-9646	-	4.7 .. 5.3 V
MC9S12DJ128	LA-9646	-	4.7 .. 5.3 V
MC9S12DJ128	LA-9646	LA-9680	4.7 .. 3.5 V
MC9S12DJ256	LA-9646	-	4.7 .. 5.3 V
MC9S12DJ256	LA-9646	LA-9680	4.7 .. 3.5 V
MC9S12DJ64	LA-9646	-	4.7 .. 5.3 V
MC9S12DJ64	LA-9646	LA-9680	4.7 .. 3.5 V
MC9S12DP256	LA-9646	-	4.7 .. 5.3 V
MC9S12DP512	LA-9646	-	4.7 .. 5.3 V
MC9S12DT128	LA-9646	-	4.7 .. 5.3 V
MC9S12DT256	LA-9646	-	4.7 .. 5.3 V
MC9S12GC128	LA-9658	-	4.7 .. 5.3 V
MC9S12GC16	LA-9658	-	4.7 .. 5.3 V
MC9S12GC32	LA-9658	-	4.7 .. 5.3 V
MC9S12GC64	LA-9658	-	4.7 .. 5.3 V

CPU	Module	Adapter	Voltage Range
MC9S12GC96	LA-9658	-	4.7 .. 5.3 V
MC9S12KG128	LA-9646	-	4.7 .. 5.3 V
MC9S12KG256	LA-9646	-	4.7 .. 5.3 V
MC9S12KG32	LA-9646	-	4.7 .. 5.3 V
MC9S12KG64	LA-9646	-	4.7 .. 5.3 V
MC9S12KT256	LA-9646	-	4.7 .. 5.3 V
MC9S12Q128	LA-9658	-	4.7 .. 5.3 V
MC9S12Q32	LA-9658	-	4.7 .. 5.3 V
MC9S12Q64	LA-9658	-	4.7 .. 5.3 V
MC9S12Q96	LA-9658	-	4.7 .. 5.3 V
MC9S12XB128	LA-9686	-	3.3 .. 5.3 V
MC9S12XB256	LA-9686	-	3.3 .. 5.3 V
MC9S12XD128	LA-9686	-	3.3 .. 5.3 V
MC9S12XD256	LA-9686	-	3.3 .. 5.3 V
MC9S12XD64	LA-9686	-	3.3 .. 5.3 V
MC9S12XDG128	LA-9686	-	3.3 .. 5.3 V
MC9S12XDP512	LA-9686	-	3.3 .. 5.3 V
MC9S12XDQ256	LA-9686	-	3.3 .. 5.3 V
MC9S12XDT256	LA-9686	-	3.3 .. 5.3 V
MC9S12XDT384	LA-9686	-	3.3 .. 5.3 V
MC9S12XDT512	LA-9686	-	3.3 .. 5.3 V
MC9S12XEG128	LA-9689	-	3.3 .. 5.3 V
MC9S12XEP100	LA-9689	-	3.3 .. 5.3 V
MC9S12XEP768	LA-9689	-	3.3 .. 5.3 V
MC9S12XEQ384	LA-9689	-	3.3 .. 5.3 V
MC9S12XEQ512	LA-9689	-	3.3 .. 5.3 V
MC9S12XET256	LA-9689	-	3.3 .. 5.3 V
MC9S12XFE128	-	-	3.3 .. 5.5 V
MC9S12XFR128	-	-	3.3 .. 5.5 V
MC9S12XS128	-	-	3.3 .. 5.5 V
MC9S12XS256	-	-	3.3 .. 5.5 V
MC9S12XS64	-	-	3.3 .. 5.5 V
MCS12KC128	LA-9646	-	4.7 .. 5.3 V
MCS12KC64	LA-9646	-	4.7 .. 5.3 V
MCS12KL128	LA-9646	-	4.7 .. 5.3 V
MCS12KL64	LA-9646	-	4.7 .. 5.3 V

Module	CPU	F-W0-10	F-W1-10	S-W0-10	S-W1-10	CHIP	TRACE	HEAD RAM
LA-9658	MC9S12C128	25.0	25.0+	16.7	25.0+	25.0		
LA-9658	MC9S12C32	25.0	25.0+	16.7	25.0+	25.0	*	
LA-9658	MC9S12C64	25.0	25.0+	16.7	25.0+	25.0		
LA-9658	MC9S12C96	25.0	25.0+	16.7	25.0+	25.0		
LA-9658	MC9S12GC128	25.0	25.0+	16.7	25.0+	25.0		
LA-9658	MC9S12GC16	25.0	25.0+	16.7	25.0+	25.0	*	
LA-9658	MC9S12GC32	25.0	25.0+	16.7	25.0+	25.0	*	
LA-9658	MC9S12GC64	25.0	25.0+	16.7	25.0+	25.0		
LA-9658	MC9S12GC96	25.0	25.0+	16.7	25.0+	25.0		
LA-9658	MC9S12Q128	25.0	25.0+	16.7	25.0+	25.0		
LA-9658	MC9S12Q32	25.0	25.0+	16.7	25.0+	25.0	*	
LA-9658	MC9S12Q64	25.0	25.0+	16.7	25.0+	25.0		
LA-9658	MC9S12Q96	25.0	25.0+	16.7	25.0+	25.0		
LA-9647	MC68HC912D60	8.0	8.0+	6.9	8.0+	8.0		
LA-9645	MC68HC912DG128	8.0	8.0+	6.9	8.0+	8.0	*	8
LA-9644	MC68HC912DT128	8.0	8.0+	6.9	8.0+	8.0	8	8
LA-9646	MC9S12A128	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12A256	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12A64	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12B128	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12B256	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12B64	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12D64	25.0	25.0+	16.7	25.0+	25.0		
LA-9657	MC9S12DB128	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12DG128	25.0	25.0+	16.7	25.0+	25.0	*	*
LA-9646	MC9S12DG256	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12DJ128	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12DJ256	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12DJ64	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12DP256	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12DP512	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12DT128	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12DT256	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12KG128	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12KG256	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12KG32	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12KG64	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MC9S12KT256	25.0	25.0+	16.7	25.0+	25.0	*	
LA-9646	MCS12KC128	25.0	25.0+	16.7	25.0+	25.0		

Module	CPU	F-W0-10	F-W1-10	S-W0-10	S-W1-10	CHIP	TRACE	HEAD RAM
LA-9646	MCS12KC64	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MCS12KL128	25.0	25.0+	16.7	25.0+	25.0		
LA-9646	MCS12KL64	25.0	25.0+	16.7	25.0+	25.0		
LA-9686	MC9S12XB128	40.0	40.0+	22.2	40.0+	40.0		
LA-9686	MC9S12XB256	40.0	40.0+	22.2	40.0+	40.0		
LA-9686	MC9S12XD128	40.0	40.0+	22.2	40.0+	40.0		
LA-9686	MC9S12XD256	40.0	40.0+	22.2	40.0+	40.0	*	
LA-9686	MC9S12XD64	40.0	40.0+	22.2	40.0+	40.0		
LA-9686	MC9S12XDG128	40.0	40.0+	22.2	40.0+	40.0		
LA-9686	MC9S12XDP512	40.0	40.0+	22.2	40.0+	40.0	*	
LA-9686	MC9S12XDQ256	40.0	40.0+	22.2	40.0+	40.0	*	
LA-9686	MC9S12XDT256	40.0	40.0+	22.2	40.0+	40.0	*	
LA-9686	MC9S12XDT384	40.0	40.0+	22.2	40.0+	40.0	*	
LA-9686	MC9S12XDT512	40.0	40.0+	22.2	40.0+	40.0	*	
LA-9689	MC9S12XEG128	40.0	40.0+	22.2	40.0+	40.0		
LA-9689	MC9S12XEP100	40.0	40.0+	22.2	40.0+	40.0		
LA-9689	MC9S12XEP768	40.0	40.0+	22.2	40.0+	40.0		
LA-9689	MC9S12XEQ384	40.0	40.0+	22.2	40.0+	40.0		
LA-9689	MC9S12XEQ512	40.0	40.0+	22.2	40.0+	40.0		
LA-9689	MC9S12XET256	40.0	40.0+	22.2	40.0+	40.0		
LA-9691	MC9S12XF128	40.0	40.0+	22.2	40.0+	40.0		
LA-9691	MC9S12XF256	40.0	40.0+	22.2	40.0+	40.0		
LA-9691	MC9S12XF384	40.0	40.0+	22.2	40.0+	40.0		
LA-9691	MC9S12XF512	40.0	40.0+	22.2	40.0+	40.0		
-	MC9S12XFE128	40.0	40.0+	22.2	40.0+	40.0		
-	MC9S12XFR128	40.0	40.0+	22.2	40.0+	40.0		
-	MC9S12XS128	40.0	40.0+	22.2	40.0+	40.0		
-	MC9S12XS256	40.0	40.0+	22.2	40.0+	40.0		
-	MC9S12XS64	40.0	40.0+	22.2	40.0+	40.0		

Probes