68kFAQ

<div align="center">

COLLABORATORS

</div>

	TITLE : 68kFAQ		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		July 28, 2024	

<div align="center">

REVISION HISTORY

</div>

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

68kFAQ

1.1 main

```
                     comp.sys.m68k FAQ List

   This is the frequently asked questions list (FAQ) for comp.sys.m68k.
   It is posted on a periodic basis to comp.sys.m68k.  This list is
   maintained by gregh@cserver.plexus.com (Greg Hawley).

   Errors, corrections, comments are encouraged.  Please send them to
   Greg Hawley at the email address listed above.  When vendor names
   or products are listed, it is not a recommendation of one vendor
   over another.  If you have a product that fits in one of the lists,
   feel free to contact the maintainer with information about the
   product.

   While a good faith effort has been made to cite from where
   information has come, some of this document may not have properly
   be attributed to its source.  Corrections should be sent to the
   maintainer.

 The following topics are covered by this FAQ.

     Cross Assemblers
     C Cross Compilers
     Monitors/Debuggers
     Data Sheets/Literature
     Samples
     Motorola BBS Access
     Getting Started Building Hardware
     Real-time Operating Systems
     S-Record Format
         Intel Hex ASCII Format
     Contributors
```

1.2 revisionhistory

```
     1.0  15FEB93  Original Release
```

```
1.1  28FEB93  Intel Hex ASCII section added, Motorola BBS #
      corrected, Some information added to Getting
      Started Building Hardware, Location of SIMTEL
      cross-assembler/simulator added.
1.1.1 08MAR93 Updated reference to Dr. Dobbs book.
```

1.3 crossassemblers

Several cross assemblers are available for Motorola 680x0
processors. Any experience that you wish to share with
the group about any of these products would be much
appreciated.

   o   a68k in source from to go with HCC see *C Cross
  Compilers* is available from the Fred Fish Amiga
  disk collection.

   o   Another version of a68k in PC executable and in
  source form? is available from the Motorola BBS.

   o   The GNU assembler GAS to go with GCC is available
  from GNU sites. See *C Cross Compilers*.

         o   A simulator/assembler for the IBM-PC compatibles on
             Simtel. The file is <msdos.crossasm>68asmsim.zip.
             The file description says "68000 cross-assembler/
             simulator with C source" See archive (location):
  swdsrv.edvz.univie.ac.at
  (/pc/dos/crossasm/68asmsim.zip), plaza.aarnet.edu.au
  (/micros/pc/oak/crossasm/68asmsim.zip), nic.switch.ch
  (/mirror/msdos/crossasm/68asmsim.zip)
  (/software/pc/simtel20/crossasm/68asmsim.zip),
  sun0.urz.uni-heidelberg.de
  (/pub/msdos/simtel/crossasm/68asmsim.zip),
  barnacle.erc.clarkson.edu
  (/pub/simtel20-cdrom/msdos/crossasm/68asmsim.zip),
  rigel.acs.oakland.edu (/pub/msdos/crossasm/68asmsim.zip),
  wuarchive.wustl.edu (/mirrors/msdos/crossasm/68asmsim.zip),
  isfs.kuis.kyoto-u.ac.jp
  (/mirrors/simtel20.msdos/crossasm/68asmsim.zip),
  ftp.uu.net
  (/systems/ibmpc/msdos/simtel20/crossasm/68asmsim.zip),
  gdr.bath.ac.uk (/simtel-cdrom/msdos/crossasm/68asmsim.zip),
  src.doc.ic.ac.uk
  (/computing/systems/ibmpc/wsmr-simtel20.army.mil/
    crossasm/68asmsim.zip)
```

## 1.4  crosscompilers

Several cross compilers are available for Motorola 680x0
processors.  Below is a list of currently know C cross
compilers.  Any comments from readers about the reliability/

portability/pricing/availability is appreciated.  What are
people using?

    o    Microtec Research has tools available for Suns,
  Ultrix/VMS, PC and possibly others.  Their address
  is 2350 Mission College Blvd., Santa Clara CA 95054.
  Phone 408-980-1300, 800-950-5554.  Fax 408-982-8266.

    o    Archemedes has unknown availablility.

    o    GNU C 2.2.2 is available in source from GNU
  anonymous FTP sites.  Some effort has gone into
  porting this code to the PC.  The PC port is
  called djgpp and is for 386 or better machines
  running DOS.  The source is free and available
  at prep.ai.mit.edu

    o    HCC available for the Amiga in the Fred Fish Amiga
  disk collections.  Fred Fish Amiga disk collection
  is archived on ux1.cso.uiuc.edu.  These archives
  are in lhz format.  The source for the utility to
  uncompress these archives is called lharc.  Lharc
  is available on comp.sources.misc archives in
  volume 11.  Try wuarchive.wustl.edu for a comp.
  sources.misc archive.

    o    A freeware C compiler is available on the Motorola
  BBS.  There is certainly a PC executable available.
  Source code, as well as executables for other
  platforms may be in the BBS.

A good comparison of supported, commercial Motorola C Cross
Compilers appeared in _Computer Design_ in January 1993 pages
103-109.  Vendors listed in this article were: 2500AD Software,
Buena Vista CO; Avocet Systems, Rockport ME; BSO/Tasking, Dedham
MA; Cygnus Support, Mountain View CA; Diab Data, Foster City CA;
Intermetrics Microsystems Software, Cambridge MA; Introl,
Milwaukee WI; Lattice, Downers Grove IL; Microtec Systems,
Santa Clara CA; Microware Systems, Des Moines IA; Oasys,
Lexington MA; Production Languages Corp (PLC), Weatherford TX;
Sierra Systems, Oakland CA; Software Development Systems
Oak Brook IL; UniPress Software, Edison NJ.

Another company that apparently sells 68k based assemblers,
cross-compilers and debuggers is Intermeterics Microsystems
Software, Inc. 733 Concord Ave., Cambridge MA 02138.

## 1.5   debuggers

One monitor available is fbug.  It is on the Motorola BBS.
The author is currently compiling and assembling it.  It
is an unsupported piece of code written by Motorola
engineers in Austin.

## 1.6   datasheets

Data sheets can be found by calling your local Motorola
Sales Office or Distributor.  Local Sales Offices and
Distributors are listed in the back of any Motorola Master
Selection Guide.  If you can not find a local
representative, Motorola Literature Distributions Center
addresses are listed by continent/region.

```
   USA
 Motorola Literature Distribution, P.O. Box 20912,
 Phoenix AZ 85036

   EUROPE
 Motorola Ltd., European Literature Center, 88 Tanners Drive,
 Blakelands, Milton Keynes, MK14 5BP England

   JAPAN
 Nippon Motorola Ltd., 4-32-1 Nishi-Gotanda, Shinagawa-ku,
 Tokyo 141 Japan

   ASIA-PACIFIC
 Motorola Semiconductors H.K. Ltd., Silicon Harbour Center,
 No. 2 Dai King Street, Tai Po Industrial Estate,
 Tai Po, N.T.,  Hong Kong
```

Ask for application notes, too. A list with Motorola document
numbers follows is found in Motorola Document BR321/D. This list
includes about 100 different documents. The author would have
typed them in, but he was short of time and space.

Programmers guide is available that describes the instructions
sets of all of the 68000-based microprocessors.

1.7   samples

For Microcomputer products try 800-845-MOTO. That is the number
for the sales division that samples microcomputers (up to
and including the 68EC040 and 68340).

1.8   motorolabbs

The Motorola BBS number is 512-891-FREE or 512-891-3733.
This is a new number perhaps less than a couple of years old.
The old number for the BBS was 512-440-2725. This number may
still work, but is not recommended. Connection to the BBS is
1200 or 2400? baud 8-N-1. For those with Internet access,
this bulletin board is mirrored on several internet
anonymous FTP sites:

```
   bode.ee.ualberta.ca
   calvin.stanford.edu
```

```
        ee.utal.edu
        red.cs.tcu.edu
        ernie.uvic.ca (login as ftp)
```

The location of the Motorola BBS information inside these
BBSs are unknown. You'll have to dig around to find them.
The site ernie.uvic.ca seems to be sanctioned by the
Motorola BBS.

In general, if you are looking for something on the network,
you can use any of a number of anonymous ftp database servers.
One of these servers is quiche.cs.mcgill. Telnet to this
machine and login as archie. From there the directions are
self-explanatory.

An anonymous ftp archive is currently being sought. Any
parties interested in helping set one up should contact
gregh@cserver.plexus.com (Greg Hawley).

1.9   gettingstarted

A percentage of the articles on comp.sys.m68k relate to
building your own hardware based on 68xxx processors. People
seem interested in building their own hardware or modifying
what others have already built. Because this is such a pop-
ular topic, no frequently asked questions list for comp.sys.
m68k would be complete without some mention of hardware hacking.
The author is in the middle of building his own 68010 box
at the writing of this FAQ.

        Choosing the Micro

One of the first decisions to be made is which processor is
to be used. Everything else falls around the choice of processors,
so this isn't as trivial a issue as it might appear. By the time
you have come to read this FAQ, you probably have joined the
Motorola 68xxx camp. Several criteria are important in the
decision of which processor to use. They are listed below.

   Criteria used to Choose a Microprocessor

   o   Availability. Where can the micro of your choice be
       bought.

   o   Cost. Can your budget afford it. When I get a
       chance I'll post what the Motorola Price List says.
       Anybody have any recommendations for sources?

   o   Package. If you don't have surface mount equipment
       in your basement, you should look for a Plastic
       Leadless Chip Carrier (PLCC), Dual In-line Package
       (DIP) or Pin Grid Array (PGA) package for the micro.
       Motorola lists part numbers that include the package
       type:

```
      MC68010L8
       \/\___/||
       |   |  |+--- Speed (in MHz)
                    |   |  +---- Package Type
                    |   |       L  - Ceramic DIP
                    |   |       LC - Ceramic DIP, Gold Lead Finish
                    |   |       P  - Plastic DIP
                    |   |       RC - Pin Grid Array, Gold Lead Finish
                    |   |       RL - Pin Grid Array, Solder Lead Finish
                    |   |       R  - Pin Grid Array, Solder Lead Finish
                    |   |       RP - Plastic Pin Grid Array
                    |   |       FN - Plastic Quad Pack (PLCC)
                    |   |      FG - Plastic Quad Plat Pack (PQFP)
                    |   |      FE - Ceramic Quad (Gull Wing)
                    |   |       FC - Plastic Quad (Gull Wing)
                    |   |
                    |   +------- Part Number
                    +----------- Motorola Prefix
```

   o On-Chip Peripherals. Many microprocessors have lots
      of on-chip stuff. Commonly available modules include
      UARTS, Timers, Chip-select generation circuitry,
      Interrupt handling circuitry, etc. Don't immediately
      write this more expensive microprocessors with on
      chip stuff. Many of the these functions are tricky
      to implement in discrete or programmable logic. The
      extra cost of having all of this extra functionality
      many be well worth your application.

   o Manufacturer. Don't forget. Motorola is not the only
      company to make 68xxx processors. At least one other
      company (and maybe others) make processors with 68xxx
      cores. Check Toshiba for information on there products.

In general, the 683xx series of embedded microcontrollers,
especially the 68340 are well suited for hardware hacking.
Byron@cc.gatech.edu (Byron A. Jeff) outlines the features of the
     68340:

   o 16 bit data bus. 32 bit address bus.
   o Quad Flat Pack (QFP) or Pin Grid Array (PGA) package.
   o HCMOS process with 90 mA load (60 uA in LPSTOP mode).
   o 2 high speed serial ports.
   o 2 general purpose timers.
   o 2 DMA channels -- max data rate of 33 Mbytes/sec.
   o watchdog and interval timers.
   o 2 eight bit parallel ports.
   o integrated IRQ, IACK and chip select circuitry.
   o On-board reset circuitry.
   o 16 MHz version available. 25 MHz version planned.
   o Background Debugging Mode.

The background debugging mode for the 68340 can be accessed
from a free debugger that runs on IBM-PC compatibles. The
debugger is available somewhere in the Motorola Freeware
BBS.

The pricing guide for Motorola entitled Motorola Semiconductor
Price List with effective date January 9, 1993 (Motorola part
number 22480-18) lists "No new designs" for the 68010. It is
unclear what the future of this part is. Be ware.

Initial Design

The best first step in designing your computer is reading the
application notes.

Glue logic

For hardware hacking perhaps the best discrete logic family to
use is HC. It doesn't use a lot of power and can be purchased
from just about anywhere. 7400, LS, ALS and other bipolar
families require higher currents. The bipolars, however, are
faster. Typically from a HC gate the propagation delay is
around 20 ns. The fast bipolars can run in the 2-3 ns range.

Don't rule out using PLDs. While it is valuable experience to
build you're whole computer out of discrete logic, if you have
access to equipment to blow PLDs, use it. PLDs keeps your board
smaller, reworks easier, and the final product more reliable.

Choosing Other Parts

DRAM controllers are available from National Semiconductors.
A DRAM Management Handbook from National explains in an
application note how to interface their controllers to
68000, 68010, 68020 and others.

Procuring Parts

Section under construction.

Software Development Package

Section under construction.

1.10   realtimeos

A freeware real-time operating system called uCOS was published
in _Embedded Systems Programming_ in May and June 1992.
Embedded Systems Programming is a Miller Freeman
Publication and can be reached at:

    Miller Freeman Inc.
    600 Harrison St.
    San Francisco CA 94107
    (415) 905-2200

       In general _Embedded Systems Programming_ is a good magazine
       for users interested in developing embedded systems. The
       magazine covers a lot of things that are relevant to programmers
       working with hardware on a more intimate level.

Real-time operating systems are also available from numerous vendors.

A book published by the publishers of Dr. Dobbs contains source code for a real-time operating system. This book contains a cross-assembler in FORTH, a BASIC interpreter and a bunch of other goodies. The reference is: _Dr. Dobbs Toolbox of 68000 Programming_, ISBN 0-13-216557-0. Unfortunately, this book is out of print. Does anyone have other appropriate references?

1.11   srecord

Chaplin@keinstr.uucp (Roger Chaplin) reposted an article written by mcdchg!motmpl!ron (Ron Widell) that explained how Motorola S-Records are formatted. This comes from a unix man page. No mention of which version of Unix is specified. This section of the FAQ is a bit long. An anonymous ftp archive is currently being sought. When one is found, the section will be placed in the archive.

```
        SREC(4)                 UNIX 5.0 (03/21/84)               SREC(4)
```

An S-record file consists of a sequence of specially formatted ASCII character strings. An S-record will be less than or equal to 78 bytes in length.

The order of S-records within a file is of no significance and no particular order may be assumed.

The general format of an S-record follow:

```
+-----------------//------------------//----------------------+
| type | count | address  |           data         | checksum |
+-----------------//------------------//----------------------+
```

```
  type       A char[2] field.  These characters describe the
             type of record (S0, S1, S2, S3, S5, S7, S8, or
             S9).
  count      A char[2] field.  These characters when paired and
             interpreted as a hexadecimal value, display the
             count of remaining character pairs in the record.

  address    A char[4,6, or 8] field.  These characters grouped
             and interpreted as a hexadecimal value, display
             the address at which the data field is to be
             loaded into memory.  The length of the field
             depends on the number of bytes necessary to hold
             the address.  A 2-byte address uses 4 characters,
             a 3-byte address uses 6 characters, and a 4-byte
             address uses 8 characters.
  data       A char [0-64] field.  These characters when paired
             and interpreted as hexadecimal values represent
             the memory loadable data or descriptive
```

information.

checksum   A char[2] field. These characters when paired and
           interpreted as a hexadecimal value display the
           least significant byte of the ones complement of
           the sum of the byte values represented by the
           pairs of characters making up the count, the
           address, and the data fields.

Each record is terminated with a line feed. If any
additional or different record terminator(s) or delay
characters are needed during transmission to the target
system it is the responsibility of the transmitting program
to provide them.

S0 Record The type of record is 'S0' (0x5330). The address

           field is unused and will be filled with zeros
           (0x0000). The header information within the data
           field is divided into the following subfields.

                    mname   is char[20] and is the
                               module name.
                    ver   is char[2] and is the
                               version number.

                    rev   is char[2] and is the
                               revision number.
                    description is char[0-36] and is a
                               text comment.

           Each of the subfields is composed of ASCII bytes
           whose associated characters, when paired,
           represent one byte hexadecimal values in the case
           of the version and revision numbers, or represent
           the hexadecimal values of the ASCII characters
           comprising the module name and description.

S1 Record The type of record field is 'S1' (0x5331). The
           address field is interpreted as a 2-byte address.
           The data field is composed of memory loadable
           data.
S2 Record The type of record field is 'S2' (0x5332). The
           address field is interpreted as a 3-byte address.
           The data field is composed of memory loadable
           data.

S3 Record The type of record field is 'S3' (0x5333). The
           address field is interpreted as a 4-byte address.
           The data field is composed of memory loadable
           data.
S5 Record The type of record field is 'S5' (0x5335). The
           address field is interpreted as a 2-byte value
           and contains the count of S1, S2, and S3 records
           previously transmitted. There is no data field.

```
S7 Record   The type of record field is 'S7' (0x5337).  The
            address field contains the starting execution
            address and is interpreted as  4-byte address.
            There is no data field.
S8 Record   The type of record field is 'S8' (0x5338).  The
            address field contains the starting execution
            address and is interpreted as  3-byte address.
            There is no data field.

S9 Record   The type of record field is 'S9' (0x5339).  The
            address field contains the starting execution
            address and is interpreted as  2-byte address.
            There is no data field.
```

EXAMPLE

Shown below is a typical S-record format file.

```
S00600004844521B
S1130000285F245F2212226A000424290008237C2A
S11300100002000800082629001853812341001813
S113002041E900084E42234300182342000824A952
S107003000144ED492
S5030004F8
S9030000FC
```

The file consists of one S0 record, four S1 records, one S5 record and an S9 record.

The S0 record is comprised as follows:

```
    S0     S-record type S0, indicating it is a header
           record.
    06     Hexadecimal 06 (decimal 6), indicating that six
           character pairs (or ASCII bytes) follow.

    00 00  Four character 2-byte address field, zeroes in
           this example.
    48     ASCII H, D, and R - "HDR".

    1B     The checksum.
```

The first S1 record is comprised as follows:
```
    S1     S-record type S1, indicating it is a data record
           to be loaded at a 2-byte address.

    13     Hexadecimal 13 (decimal 19), indicating that
           nineteen character pairs, representing a 2 byte
           address, 16 bytes of binary data, and a 1 byte
           checksum, follow.
    00 00  Four character 2-byte address field; hexidecimal
           address 0x0000, where the data which follows is to
           be loaded.

    28 5F 24 5F 22 12 22 6A 00 04 24 29 00 08 23 7C Sixteen
           character pairs representing the actual binary
           data.
```

```
    2A      The checksum.
```

The second and third S1 records each contain 0x13 (19)
character pairs and are ended with checksums of 13 and 52,
respectively. The fourth S1 record contains 07 character
pairs and has a checksum of 92.

The S5 record is comprised as follows:

```
    S5      S-record type S5, indicating it is a count record
            indicating the number of S1 records.



    03      Hexadecimal 03 (decimal 3), indicating that three
            character pairs follow.

    00 04   Hexadecimal 0004 (decimal 4), indicating that
            there are four data records previous to this
            record.
    F8      The checksum.
```

The S9 record is comprised as follows:

```
    S9      S-record type S9, indicating it is a termination
            record.
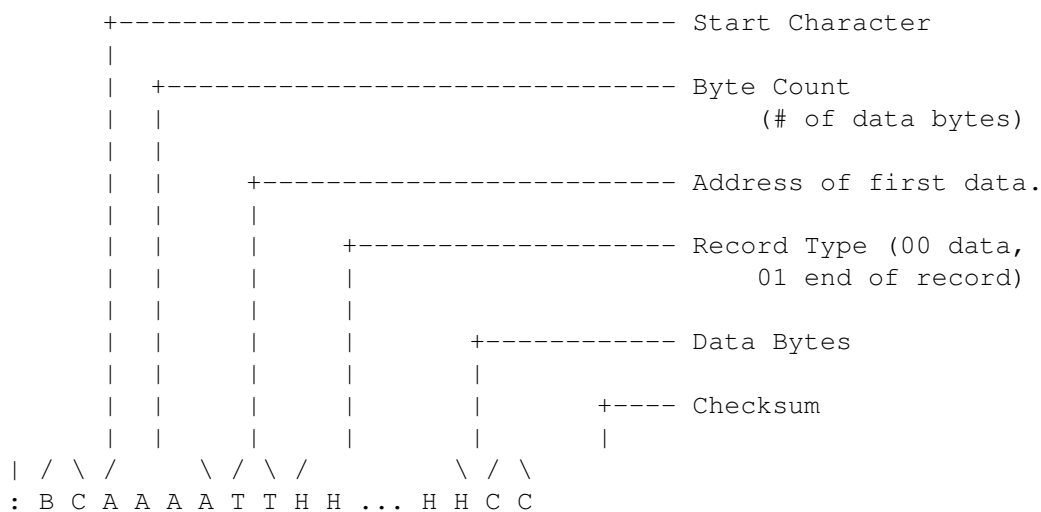    03      Hexadecimal 03 (decimal 3), indicating that three
            character pairs follow.

    00 00   The address field, hexadecimal 0 (decimal 0)
            indicating the starting execution address.
    FC      The checksum.
```

1.12 intelhex

Intel HEX-ASCII format takes the form:

```
        +--------------------------------- Start Character
        |
        |  +------------------------------ Byte Count
        |  |                                   (# of data bytes)
        |  |
        |  |     +------------------------ Address of first data.
        |  |     |
        |  |     |    +------------------- Record Type (00 data,
        |  |     |    |                         01 end of record)
        |  |     |    |
        |  |     |    |      +----------- Data Bytes
        |  |     |    |      |
        |  |     |    |      |       +---- Checksum
        |  |     |    |      |       |
   | / \ /    \ / \ /        \ / \
   : B C A A A A T T H H ... H H C C
```

An examples:

```
    :10000000DB00E60F5F1600211100197ED300C3004C
    :10001000000000101030307070F0F1F1F3F3F7F7FF2
    :01002000FFE0
    :00000001FF
```

This information comes from _Microprocessors and Programmed
Logic_, Second Edition, Kenneth L. Short, 1987, Prentice-Hall,
ISBN 0-13-580606-2.

Provisions have been made for data spaces larger than 64 kBytes.
The above reference does not discuss them. I suspect there is
a start of segment type record, but I do not know how it is
implemented.

1.13   contributors

     Many people have contributed to this list. Below is a list
     of people who have helped by either their direct input or through
their postings to comp.sys.m68k. I can't list everyone, but I
have tried to include many.

```
    walvdrk_r@research.ptt.nl (Kees van der Wal)
    byron@cc.gatech.edu (Byron A Jeff)
          mcdchg!motmpl!ron (Ron Widell)
    rrt@mpd.tandem.com (Bob Teisberg)
    benstn@olivetti.nl (Ben Stuyts)
    csuley@cs.cornell.edu (Christopher Suley)
    idr@mailhost.cs.pdx.edu (Ian D Romanick)
    wayne@netcom.com (wayne t. watson)
```