Introduction to ARM machine description for GCC

László Vidács, lac@rgai.hu

Last updated 17 February 2003
version 0.1

Preamble

This document is an introduction to GCC machine descriptions for ARM processor arhitecture. After a short overview of compilation process and the Register Transfer Language (RTL), the parts of a machine description (md) are introduced. First the ARM `md` directory is described, after which more about the `arm.md` file can be read. For details see GNU manual [1].

Contents

1 Compilation in GCC

In the GCC compiler, on general level the following passes are invoked during the compilation of a source file, for details see [1].
1. Parsing. This pass is invoked only once, the whole input is parsed and a high level tree representation is generated from it (one function at a time).
2. RTL generation. The tree code is transformed into intermediate code called RTL (Register Transfer Language).
3. RTL–RTL optimizations. Several optimization passes including common subexpression elimination, loop optimization, data flow analysis, instruction scheduling, register allocation, basic block reordering, jump optimization, machine dependent optimizations, etc.
4. Output generation. This is the final pass, outputs the assembly code generated from RTL.
The assembly code is the output of GCC, an assembler is used to produce the binary object file.

2 Register Transfer Language

The intermediate representation of program code is called RTL. RTL is an abstract description of a program, parameterized by the machine description. It has two forms: internal form with structures, and textual form used in machine descriptions and debug dumps. The textual form is a LISP-like representation composed of embedded lists. There are 5 rtl *object types*: integer, wide integer, string, vector and expression. Strings can be written in both forms: `"string"` and `{string}`; this is useful when C code is embedded. The written form of the vector is a square bracket surrounding the elements (`[ ... ]`). Expressions are classified with expression codes. Possible expression codes and their meanings are machine dependent and are defined in the file *rtl.def*.
A *machine mode* describes a size of data object and the representation used for it. Modes are written as *XX*`mode`, where *XX* can be `QI` (quarter integer), `SI` (single integer), `HF` (half floating) and so on. For example `(mem:SI ...)` is a Single Integer mode expression.
There are rtl expressions for
- registers and memory – `(reg:m number)`
- arithmetic – `(plus:m x y)`
- comparisons – `(eq:m x y)`, conversions – `(sign_extend:m x)`
- calls – `(call (mem:fm addr) nbytes)`

and others.

Instructions are also RTL expressions, called *insn*. Every insn has one of the following 6 expression codes: insn, jump_insn, call_insn, code_label, barrier, note.

3 Source directory of ARM target

There is a separate directory for various machine descriptions in gcc source tree. The three necessary files in a machine description directory are *machine*.h, *machine*.c, *machine*.md. The *machine*.h contains macro definitions for the target processor (processor modes, register numbering and usage) and memory usage, declarations of functions defined in *machine*.c. The *machine*.md is written in RTL (Register Transfer Language), and uses functions and defined values from *machine*.h and *machine*.c. It contains patterns for generating RTL instructions from the parse tree, and for generating final assembler code after RTL-to-RTL optimization passes.
The directory gcc/config/arm/ contains sources describing machine specific information for ARM processors, for both ARM and THUMB instruction sets. In addition to arm.c, arm.h and arm.md there are some other files in this directory, which are also presented in this section.

3.1 Structure of arm.h

File arm.h contains macro definitions and some enum definitions for the following purposes:
o Run-time Target Specification
Determines the arm cpu type (arm2, arm6, arm7m, arm7tdmi ...). The value of EXTRA_SPECS macro is added to the gcc specs file. Target switches, flags and options: floating point unit type, arm extensions, ARM or THUMB code, etc.
o Target machine storage layout
Endianism; bits per word; pointer size; stack, function boundary,...
o Standard register usage
Register allocation: fixed, call used registers, conditional register usage, reg. allocation order, ...
o Register and constant classes
Registers are divided into register classes: fpu, lo, hi, stack, base, cc, general. Macros for register allocation, constant pool addressing.
o Stack layout; function entry, exit and calling
o Addressing modes
o Position Independent Code

- Pragmas for compatibility with Intel's compilers
- Constant pool machine dependent reorganization
- Branch elimination
- Codes that are matched by predicates in `arm.c`

3.2 Structure of `arm.c`

File `arm.c` contains c routines that support patterns in `arm.md` (optimization, output, etc.):

- initializing the GCC target structure
- condition codes, arm chips, cores
- exception types, instruction types
- pragma calls
- machine attributes
- coping with patterns (`match_operator`, `match_operand`, ...)
- RTL generation routines
- writing debug information
- constant pool manipulating
- output assembly
- function prologue/epilogue codes

3.3 Structure of `arm.md`

Described in Section 4.

3.4 Files controlling the compilation of `md`

During the execution of GCC configure script, it can incorporate makefile fragments into the Makefile from the target config directory. Fragments named `t-`*`target`* are used to set Makefile parameters for configuring GCC to produce code for specific target of ARM architecture. In target fragments we can control: libgcc2 flags, floating point emulation, crti flags, multilib options and other parameters. The following `t-`*`target`* files are in the ARM config directory:

t-arm-aout	t-semi
t-arm-coff	t-strongarm-coff
t-arm-elf	t-strongarm-elf
t-linux	t-strongarm-pe
t-netbsd	t-xscale-coff
t-pe	t-xscale-elf
t-riscix	

3.5 Target machine variations

Variations of target machine descriptions are also located in this directory for various object formats, operating systems and architectures.
The configure script chooses the suitable files (see `gcc-ver/configure.in`):

aof.h	Advanced RISC Machines, ARM compilation, AOF Assembler
aout.h	ARM with a.out
arm-wince-pe.h	ARM with PE obj format running under the WinCE operating system
coff.h	ARM with COFF object format
conix-elf.h	ARM with ConiX OS
ecos-elf.h	Ecos based ARM systems using ELF
elf.h	ARM with ELF obj format
Freebsd.h	StrongARM running FreeBSD using the ELF format
linux-elf.h	ARM running Linux-based GNU systems using ELF
linux-gas.h	ARM Linux-based GNU systems version
Netbsd.h	NetBSD/arm a.out version
pe.c	ARM/pe
pe.h	ARM with PE obj format
riscix.h	ARM RISCiX version
riscix1-1.h	ARM RISCiX 1.1x version
rix-gas.h	ARM RISCiX(stabs) version
rtems-elf.h	RTEMS based ARM systems using ELF
semi.h	ARM on semi-hosted platform
semiaof.h	ARM on semi-hosted platform AOF Syntax assembler
strongarm-coff.h	StrongARM systems using COFF

`strongarm-elf.h`	non-Linux based StrongARM systems using ELF
`strongarm-pe.h`	ARM with PE obj format
`uclinux-elf.h`	ARM running ucLinux using ELF
`unknown-elf-oabi.h`	non-Linux based ARM systems using ELF old ABI
`unknown-elf.h`	non-Linux based ARM systems using ELF
`vxarm.h`	ARM with targetting the VXWorks run time environment
`xscale-coff.h`	Xscale systems using COFF
`xscale-elf.h`	Xscale architectures using ELF arm.c

3.6 Miscellaneous files

`arm-protos.h`	prototypes for exported functions defined in arm.c and pe.c
`crti.asm`, `crtn.asm`	.fini and.init sections (used before compiled program starts and terminates)
`lib1funcs.asm`	libgcc routines for ARM cpu: division routines

4 ARM machine description

The file `arm.md` is divided into 3 sections: constants, attributes and insn patterns.

4.1 Constants

Constant definitions are for legibility and better maintenance. A constant definition is a vector of name-value pairs, each constant appears in generated file `insn-constants.h` as a `#define`.
There are two types of constants in ARM md.

1. Register numbers
This section gives a register number for special arm registers. According to ARM 7TDMI Data Sheet [2], there are 17 registers in the processor's System & User mode in textttARM state. Registers numbered from 0 to 13 are general registers. R14 receives a copy of R15 when BL (branch and Link) is executed, R15 holds the PC (program counter) and R16 is the CPSR

(Current Program Status Register).
In `arm.md` the following registers have constant numbers: scratch reg, stack pointer, return address reg, program counter, condition code pseudo reg.

```
;; Register numbers
(define_constants
  [(IP_REGNUM     12) ; Scratch register
   (SP_REGNUM     13) ; Stack pointer
   (LR_REGNUM     14) ; Return address register
   (PC_REGNUM     15) ; Program counter
   (CC_REGNUM     24) ; Condition code pseudo register
   (LAST_ARM_REGNUM 15)
  ]
)
```

2. Machine specific operations
In md we can define machine specific instructions. Each instruction has its own constant, this can be defined with pattern `unspec_*` or `unspec_volatile` (the second is used for defining volatile operations and operations that may trap). These codes may appear inside a pattern of an insn, inside a parallel, or inside an expression. Example from `arm.md`:

```
(define_constants
  [(UNSPEC_SIN 0) ; 'sin' operation (MODE_FLOAT):
   (UNPSEC_COS 1)\tab  ; 'cos' operation (MODE_FLOAT):
   (UNSPEC_PUSH_MULT 2) ; 'push multiple' operation:
...
]
)
```

4.2 Attributes and function units

4.2.1 Attributes

Instructions can have one or more *attributes*. For each attribute we have to define a set of values. Every generated insn is assigned a value for each attribute. They are defined with pattern define_attr of the following form:
(define_attr name list-of-values default)

Example from `arm.md`: instruction length, constant pool range:

```
; LENGTH of an instruction (in bytes)
(define_attr "length" "" (const_int 4))
; POOL_RANGE is how far away from a constant pool entry that this
; insn can be placed. If the distance is zero, then this insn will
```

```
; never reference the pool.
; NEG_POOL_RANGE is nonzero for insns that can reference a constant
; pool entry before its address.
(define_attr "pool_range" "" (const_int 0))
(define_attr "neg_pool_range" "" (const_int 0))
```

In `arm.md` there are various types of attributes describing:
Instruction type, constant pool range, scheduling, condition codes, etc.

4.2.2 Function units (instruction scheduling)

On most RISC machines, there are instructions whose results are not available for a specific number of cycles. Common cases are instructions that load data from memory.
For each instruction we can define the amount of time that must elapse between the execution of instruction and the time its result can be used. A machine is divided into *function units*, each of which executes a specific class of instructions in first-in-first-out order. Function units can be defined with define_function_unit:
(define_function_unit {*name*} {*num-units*} {*n-users*} {*test*}
{*ready-delay*} {*issue-delay*} *[*{*conflict-list*}*])*
Typical uses are where a floating point function unit can pipeline either single- or double-precision operations, but not both. In `arm.md` there are function units defined for the floating point unit, write buffer, write blockage unit and core unit.
An example from core unit:

```
; Everything must spend at least one cycle in the core unit
(define_function_unit "core" 1 0
  (eq_attr "core_cycles" "single") 1 1)
```

4.3 Insn patterns

This is the most important and largest part of `arm.md`. There are several types of patterns which are used for different purposes: `define_insn`, `define_expand`, `define_split`, `define_insn_and_split`, `define_peephole`, `define_peephole2`, etc. Of these the `define_insn` is the most important, which is described in the following.

4.3.1 Insn pattern format

Insn patterns are defined with `define_insn` expressions, which have four or five operands according to the following form:

(define_insn {name} RTL_template
condition output_template {attribute_values})

1. name – a name from the insn standard names list for patterns of RTL generation (2nd pass) and no name for patterns of assembler code generation (if a name is marked with * then it is treated as a namesless one and can be used for debugging).
2. RTL template – a vector of incomplete RTL expressions which show what the instruction should look like.
It may contain `match_operand`, `match_operator` expressions that stand for operands of instruction. There may be constraints for operands.
3. condition – a C expression for testing whether the insn body matches the pattern. For named patterns the condition may depend only on target-machine-type flags.
4. output template – a string that says how to output assembler code for this insn. `%` markes the position where the value of operands are substituted.
5. attribute values – optional, the values of attributes for this insn.

For example the ARM state mov instruction for single integers:

```
1.  (define_insn "*arm_movsi_insn"
2.  [(set (match_operand:SI 0 "nonimmediate_operand" "=r,r,r, m")
3.  (match_operand:SI 1 "general_operand" "rI,K,mi,r"))]
4.  "TARGET_ARM
5.  && ( register_operand (operands[0], SImode)
6.  || register_operand (operands[1], SImode))"
7.  "@
8.  mov%?\\t%0, %1
9.  mvn%?\\t%0, #%B1
10.  ldr%?\\t%0, %1
11.  str%?\\t%1, %0"
12.  [(set_attr "type" "*,*,load,store1")
13.  (set_attr "predicable" "yes")
14.  (set_attr "pool_range" "*,*,4096,*")
15.  (set_attr "neg_pool_range" "*,*,4084,*")]
16.  )
```

The example has the following meaning:

Line	Comment
`1.`	instruction name
`2-3.`	RTL template instruction pattern: (*match_operand n predicate constraint*) is a placeholder of operand number n, predicate is a function, if it returns 0 then the pattern fails to match, constraint controls the reload to choose the best register class for a value.
`4-6.`	Condition: This is a string which contains a C expression that is the final test to decide whether an insn body matches this pattern.
`7-11.`	Output Template: the assembler code that this insn generated in the final pass, `%n` is the operand number n.
`12-15.`	Attributes: sets the value of attributes

4.3.2 Pattern types

There are six types of insn patterns in `arm.md`:

1. define_insn

This pattern is used in two passes:

• in the RTL-generation pass the compiler chooses only from the named patterns (standard names) to generate the initial RTL code from AST. In this pass the compiler chooses the pattern with the corresponding name, the RTL template and the constraints are ignored.

• in the output generation, the compiler emits the assembly code using unnamed and named patterns (the name is not significant in this stage)

2. define_expand

This pattern has the following three purposes:

1. `DONE` is present in the output template: enables the user to manually create new insns for the insn list (function `emit_insn()`)

2. `FAIL` is present in the output template: tells the compiler that this is wrong way, choose another pattern

3. none of the above: behaves like `define_insn` pattern (this type also takes part in the RTL generation)

3. define_split

Splits the pattern into multiple insns during the RTL optimizations, the aim is to enhance sheduling, or to cope with complex expressions.

4. define_insn_and_split

Shorter form, it can be used when the `define_insn` pattern exactly matches the `define_split pattern`.

5. define_peephole

Peephole optimization runing at RTL to assembly output time, deprecated

according to the GNU manual [1]. In `arm.md` used only in Misc patterns, there are 13 peephole for extended post-inc insns and possible load- and store-multiple insns.

6. define_peephole2

Used in RTL to RTL optimization stage 24, after register allocations. There are 5 `define_peephole2`s: addition patterns (3), boolean (1), misc (1).

4.3.3 Structure of file

Insn patterns in `arm.md` are groupped by the type of the instructions:
Addition insns, Multiplication insns
Division insns
Modulo insns
Boolean and,ior,xor insns
Minimum and maximum insns
Shift and rotation insns
Unary arithmetic insns
Fixed <–> Floating conversion insns
Zero and sign extension instructions.
Move insns (including loads and stores)
load- and store-multiple insns
Compare & branch insns
Conditional branch insns
Scc insns
Conditional move insns
Jump and linkage insns
Misc insns: combination of arithmetic, cond code and shifts, pre-inc instructions, extended post-inc expressions, special patterns for dealing with the constant pool, miscellaneous `THUMB` patterns

5 GNU Free Documentation License

Version 1.2, November 2002
Copyright (C) 2000,2001,2002 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.
0. PREAMBLE
The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.
This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.
We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.
1. APPLICABILITY AND DEFINITIONS
This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.
A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.
A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall

subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.
The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.
The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.
A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".
Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.
The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along

with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.
It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.
4. MODIFICATIONS
You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:
A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section Entitled "History", Preserve its Title, and add to it

an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
O. Preserve any Warranty Disclaimers.
If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.
You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.
You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher

that added the old one.
The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.
5. COMBINING DOCUMENTS
You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.
The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.
In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".
6. COLLECTIONS OF DOCUMENTS
You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.
You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.
7. AGGREGATION WITH INDEPENDENT WORKS
A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.
If the Cover Text requirement of section 3 is applicable to these copies of

the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

5.1 ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:
Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".
If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:
with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.
If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

References

[1] Richard M. Stallman: GNU Compiler Collection Internals, A GNU Manual, FSF, 2002

[2] ARM 7TDMI Data Sheet, Advanced RISC Machines Ltd, 1995