CoreMark 1.0 - STM32 – Testbericht

Die STM32-Mikrocontrollerfamilie ist eine weit verbreitete Plattform für komplexe Mikrocontrolleranwendungen.
Laut eigenen Angaben hatte ST Microelectronics schon 2010 45% Marktanteil bei den Cortex-M Mikrocontrollern für Anforderungen die zwischen 8bit-Controllern und Embedded PCs liegen.
Bei www.harerod.de werden die STM32 aktuell für die meisten Projekte eingesetzt.

Üblicherweise erfolgt die Bausteinauswahl schwerpunktmäßig über die benötigte Peripherie. In diesem Bericht soll jedoch die Leistung der ARM Cortex-M Prozessoren für verschiedene Compilersettings und Systemeinstellungen untersucht werden.

Für diesen Zweck wurde CoreMark von coremark.org / eembc.org in die in die CooCox/GCC Umgebung portiert. CoreMark kann nach Registrierung kostenlos heruntergeladen werden.

What is CoreMark?
Processors and associated systems are getting increasingly complex requiring increasingly complex benchmarks to analyze. The current and future EEMBC benchmarks are aimed at specific embedded market segments and are very successful at approximating real-world performance of embedded devices. However, there is also a need for a widely-available, generic benchmark specifically targeted at the processor core. Introducing CoreMark -- Developed by EEMBC, this is a simple, yet sophisticated, benchmark that is designed specifically to test the functionality of a processor core. Running CoreMark produces a single-number score allowing users to make quick comparisons between processors.

Gemäß den CoreMark-Run-Rules wurde die MCU jeweils vom Framework in die gewählte Betriebsart gebracht. Danach wurde die main-Funktion von CoreMark ausgeführt und deren Ergebnisse protokolliert.

Topic: Run rules - What is and is not allowed.
Required:
1 - The benchmark needs to run for at least 10 seconds.
2 - All validation must succeed for seeds 0,0,0x66 and 0x3415,0x3415,0x66,
buffer size of 2000 bytes total.
o If not using command line arguments to main:
> make XCFLAGS="-DPERFORMANCE_RUN=1" REBUILD=1 run1.log
> make XCFLAGS="-DVALIDATION_RUN=1" REBUILD=1 run2.log
3 - If using profile guided optimization, profile must be generated using seeds of 8,8,8,
and buffer size of 1200 bytes total.
> make XCFLAGS="-DTOTAL_DATA_SIZE=1200 -DPROFILE_RUN=1" REBUILD=1 run3.log
4 - All source files must be compiled with the same flags.
5 - All data type sizes must match size in bits such that:
o ee_u8 is an 8 bits datatype. o ee_s16 is an 16 bits datatype.
o ee_u16 is an 16 bits datatype. o ee_s32 is an 32 bits datatype.
o ee_u32 is an 32 bits datatype.
Allowed:
- Changing number of iterations
- Changing toolchain and build/load/run options
- Changing method of acquiring a data memory block
- Changing the method of acquiring seed values
- Changing implementation in core_portme.c
- Changing configuration values in core_portme.h
- Changing core_portme.mak
Not allowed:
- Changing of source file other then core_portme* (use make check to validate)

Getestete MCUs

STM32F407VG ARM Cortex-M4
MCU: 512kB Flash, 128kB+64kB SRAM, 168MHz Cortex-M4

STM32F103R8 ARM Cortex-M3
MCU: 64kB Flash, 20kB SRAM, 72MHz Cortex-M3

Legende

Adaptive Real Time Accelerator / Flash Einstellungen:
- xWS – x=Anzahl Waitstates
- P – Instruction Prefetch
- D- Data Management
- I – Instruction Cache Memory

Leistungsaufnahme:
Die MCUs wurden mit 3,3V versorgt. Es wurde der Gesamtstrom der digitalen und analogen MCU-Versorgung gemessen. Der Active Current wurde während der CoreMark-Berechnung gemessen, der Idle Current in einer while(1)-Endlosschleife.

STM32F407VG: Vergleich der Compileroptimierungsstufen und Compilerversionen beim GCC

Dieser Test zeigen deutliche Unterschiede zwischen GCC 4.8.3 und GCC 4.6.2. Während die ältere Version das beste Ergebnis in –O2 erreichte, bringt die 4.8.3 in –O3 ihre höchste CoreMark. Allenfalls wegen der Codegröße lohnt sich ein Blick auf andere Optimierungsstufen. Seit Version 4.8 gibt es die Optimierungsstufe –Og, welche für Debugging optimierten Code erzeugt.

MCU	Compiler	Optimizer	Codesize text [kB]	Code size data [kB]	Code size bss [kB]	Clock [MHz]	Flash	Core Mark	Core Mark / MHz	Voltage Range	MCU Current Active [mA]	MCU Current Idle [mA]	Core Mark/ mA	Core Mark/ mW	µW / Core Mark
STM32F407VG	GCC4.8.3	-O3	17648	44	8220	168	5WS-I-D-P	373,1	2,22	2,7..3,6V	42,27	34,91	8,83	29,13	34,33
STM32F407VG	GCC4.8.3	-O2	13556	44	8220	168	5WS-I-D-P	354,6	2,11	2,7..3,6V	42,14	35,18	8,42	27,77	36,01
STM32F407VG	GCC4.8.3	-O1	13744	44	8220	168	5WS-I-D-P	315,7	1,88	2,7..3,6V	43,50	54,55	7,26	23,95	41,75
STM32F407VG	GCC4.8.3	-O0	20852	164	8220	168	5WS-I-D-P	96,2	0,57	2,7..3,6V	43,27	50,00	2,22	7,34	136,26
STM32F407VG	GCC4.8.3	-Os	12089	44	8220	168	5WS-I-D-P	299,2	1,78	2,7..3,6V	43,09	34,18	6,94	22,91	43,65
STM32F407VG	GCC4.8.3	-Og	13892	44	8220	168	5WS-I-D-P	258,4	1,54	2,7..3,6V	42,18	35,23	6,13	20,22	49,47
STM32F407VG	GCC4.6.2	-O3	25300	44	8220	168	5WS-I-D-P	345,8	2,06	2,7..3,6V	45,32	49,09	7,63	25,18	39,71
STM32F407VG	GCC4.6.2	-O2	17624	44	8220	168	5WS-I-D-P	365,0	2,17	2,7..3,6V	44,00	47,73	8,29	27,37	36,53
STM32F407VG	GCC4.6.2	-O1	15424	44	8220	168	5WS-I-D-P	283,9	1,69	2,7..3,6V	43,05	34,18	6,60	21,76	45,95
STM32F407VG	GCC4.6.2	-O0	24132	164	8220	168	5WS-I-D-P	95,7	0,57	2,7..3,6V	43,82	52,73	2,18	7,20	138,82
STM32F407VG	GCC4.6.2	-Os	12277	44	8220	168	5WS-I-D-P	292,0	1,74	2,7..3,6V	40,77	34,55	7,16	23,63	42,32

Abbildung 1 - STM32F407VG: Vergleich der Compileroptimierungsstufen und Compilerversionen

STM32F407VG: Vergleich der ART- und Flasheinstellungen

Bei der vorliegenden Betriebsart (Versorgungsspannung) taktet der Flash-Speicher mit 30MHz. Daher werden Waitstates je nach Prozessorgeschwindigkeit fällig, die teilweise über größere Lesebreiten und Caches ausgeglichen werden.
Der ART verspricht laut Datenblatt effektiven 0WS-Betrieb bei 168MHz. Dieses Versprechen wird gehalten, wenn man die CoreMark/MHz zwischen 168MHz und 30MHz Prozessortakt vergleicht.

MCU	Compiler	Optimizer	Codesize text [kB]	Code size data [kB]	Code size bss [kB]	Clock [MHz]	Flash	Core Mark	Core Mark / MHz	Voltage Range	MCU Current Active [mA]	MCU Current Idle [mA]	Core Mark / mA	Core Mark / mW	µW / Core Mark
STM32F407VG	GCC4.8.3	-O3	17648	44	8220	168	5WS-I-D-P	373,1	2,22	2,7..3,6V	42,27	34,55	8,83	29,13	34,33
STM32F407VG	GCC4.8.3	-O3	17648	44	8220	168	5WS-I-D	370,4	2,20	2,7..3,6V	40,91	34,55	9,05	29,88	33,47
STM32F407VG	GCC4.8.3	-O3	17648	44	8220	168	5WS-D	180,0	1,07	2,7..3,6V	47,27	34,73	3,81	12,56	79,60
STM32F407VG	GCC4.8.3	-O3	17648	44	8220	168	5WS-I	370,4	2,20	2,7..3,6V	41,00	34,68	9,03	29,81	33,55
STM32F407VG	GCC4.8.3	-O3	17648	44	8220	168	5WS-P	214,3	1,28	2,7..3,6V	51,36	34,68	4,17	13,77	72,64
STM32F407VG	GCC4.8.3	-O3	17648	44	8220	168	5WS	180,0	1,07	2,7..3,6V	46,82	34,55	3,84	12,68	78,83
STM32F407VG	GCC4.8.3	-O3	17648	44	8220	30	0WS-I-D-P	67,0	2,23	2,7..3,6V	12,09	9,09	5,54	18,30	54,66
STM32F407VG	GCC4.8.3	-O3	17648	44	8220	30	0WS-I-D	67,0	2,23	2,7..3,6V	10,14	9,09	6,61	21,82	45,82
STM32F407VG	GCC4.8.3	-O3	17648	44	8220	30	0WS-D	67,0	2,23	2,7..3,6V	18,05	9,09	3,71	12,26	81,58
STM32F407VG	GCC4.8.3	-O3	17648	44	8220	30	0WS-I	67,0	2,23	2,7..3,6V	10,09	8,95	6,64	21,92	45,62
STM32F407VG	GCC4.8.3	-O3	17648	44	8220	30	0WS-P	67,0	2,23	2,7..3,6V	20,77	9,09	3,23	10,65	93,91
STM32F407VG	GCC4.8.3	-O3	17648	44	8220	30	0WS	67,0	2,23	2,7..3,6V	18,00	9,09	3,72	12,29	81,37

Abbildung 2 - STM32F407VG: Vergleich der ART- und Flasheinstellungen

STM32F407VG: Vergleich der Compileroptimierungsstufen beim IAR EWARM-CD-7101-6735

Zum Abschluss noch ein Vergleichstest mit dem IAR Compiler. Im Gegensatz zum GCC gibt es hier mehrere Versionen, u.a. eine Gratisversion mit 32kB Codelimitierung (die hier getestet wird) und kostenpflichtige Versionen.
Sowohl Codesize als auch CoreMark dieses dedizierten Mikrocontroller-Compilers beeindrucken bei diesem Test.

MCU	Compiler	Optimizer	Codesize text [kB]	Code size data [kB]	Code size bss [kB]	Clock [MHz]	Flash	Core Mark	Core Mark / MHz	Voltage Range	MCU Current Active [mA]	MCU Current Idle [mA]	Core Mark/ mA	Core Mark/ mW	µW / Core Mark
STM32F407VG	IAR	-O High Size	12016	356	8364	168	5WS-I-D-P	313,5	1,87	2,7..3,6V	43,41	31,27	7,22	23,83	41,96
STM32F407VG	IAR	-O High Speed - No Size Constraints	16790	358	8364	168	5WS-I-D-P	555,8	3,31	2,7..3,6V	45,41	30,82	12,24	40,39	24,76
STM32F407VG	IAR	-O High Speed	13982	358	8364	168	5WS-I-D-P	427,7	2,55	2,7..3,6V	42,18	31,14	10,14	33,46	29,89
STM32F407VG	IAR	-O High Balanced	12474	358	8364	168	5WS-I-D-P	358,9	2,14	2,7..3,6V	41,55	31,82	8,64	28,50	35,08
STM32F407VG	IAR	-O Medium	12460	444	8372	168	5WS-I-D-P	265,3	1,58	2,7..3,6V	41,09	34,18	6,46	21,30	46,94
STM32F407VG	IAR	-O Low	13228	1576	8372	168	5WS-I-D-P	214,1	1,27	2,7..3,6V	42,00	56,82	5,10	16,82	59,44
STM32F407VG	IAR	-O None	14092	1576	8372	168	5WS-I-D-P	187,5	1,12	2,7..3,6V	42,09	33,64	4,45	14,70	68,03

Abbildung 3 - STM32F407VG: Vergleich der Compileroptimierungsstufen beim IAR EWARM-CD-7101-6735

STM32F103R8: Vergleich der ART- und Flasheinstellungen

Bei der vorliegenden Betriebsart (Versorgungsspannung) taktet der Flash-Speicher mit 24MHz. Daher werden je nach Prozessorgeschwindigkeit Waitstates fällig, die teilweise über Caches ausgeglichen werden.
Dieser MCU fehlt jedoch der ART, stattdessen steht ein Prefetch-Verfahren zur Verfügung.
Interessant ist hier zu sehen, dass das Prefetching die Waitstates nur teilweise ausgleichen kann, so dass die CoreMark/MHz mit steigendem Prozessortakt von 2,19 nach 1,59 sinkt.
Vergleicht man die CoreMark/MHz zwischen Cortex-M3 und Cortex-M4 bei 0WS, so zeigen sich für diesen Test keine wesentlichen Unterschiede.

MCU	Compiler	Optimizer	Codesize text [kB]	Code size data [kB]	Code size bss [kB]	Clock [MHz]	Flash	Core Mark	Core Mark / MHz	Voltage Range	MCU Current Active [mA]	MCU Current Idle [mA]	Core Mark/ mA	Core Mark/ mW	µW / Core Mark
STM32F103R8	GCC4.8.3	-O3	18972	48	8224	72	2WS-P	114,6	1,59	2,7..3,6V	29,68	23,91	3,86	12,74	78,48
STM32F103R8	GCC4.8.3	-O3	18972	48	8224	72	2WS	83,8	1,16	2,7..3,6V	27,18	24,59	3,08	10,18	98,24
STM32F103R8	GCC4.8.3	-O3	18972	48	8224	48	1WS-P	92,1	1,92	2,7..3,6V	23,91	20,59	3,85	12,72	78,64
STM32F103R8	GCC4.8.3	-O3	18972	48	8224	48	1WS	78,7	1,64	2,7..3,6V	22,64	20,95	3,48	11,47	87,18
STM32F103R8	GCC4.8.3	-O3	18972	48	8224	24	0WS-P	52,5	2,19	2,7..3,6V	15,23	8,86	3,45	11,37	87,92
STM32F103R8	GCC4.8.3	-O3	18972	48	8224	24	0WS	52,5	2,19	2,7..3,6V	15,64	16,59	3,36	11,08	90,29

Abbildung 4 - STM32F103R8: Vergleich der ART- und Flasheinstellungen

Fazit

Zu Beginn aller Mikrocontroller-Projekte steht die Auswahl der MCU und dazu passender Entwicklungsumgebung. Das vorliegende Papier entstand, um bereits publizierte Testergebnisse zu überprüfen (coremark.org) und eine Entscheidungsgrundlage für Prozessorkonfiguration und Compilereinstellungen zu bekommen.
Auf Kundenseite besteht üblicherweise der Wunsch, schon bekannte Bausteine und Compiler einzusetzen. Sollte insbesondere der zweite Parameter frei sein, so fällt die Wahl gerne auf ein ohne Lizenzierungsverfahren frei verfügbares Werkzeug.

Danksagung

Danke an EEMBC, für CoreMark. Danke an STM und ARM für die MCU-Familie. Danke an IAR für die 32kB-Teststellung. Danke ans ARM-GCC-Team, für den freien Compiler.

Quellen

STM32F103x8:
http://www.st.com/web/en/resource/technical/document/datasheet/CD00161566.pdf

STM32F40xx:
http://www.st.com/web/en/resource/technical/document/datasheet/DM00037051.pdf
STM32F405xx/07xx, STM32F415xx/17xx, STM32F42xxx and

STM32F43xxx advanced ARM-based 32-bit MCUs Reference Manual:
http://www.st.com/web/en/resource/technical/document/reference_manual/DM00031020.pdf

www.coremark.org