ANDES TECHNOLOGY

HelloGCC 2013

Port GCC to a new architecture
Case study: ***nds32***

2013.11.16

Chung-Ju Wu (吳中如)

Overview of Andes Technology

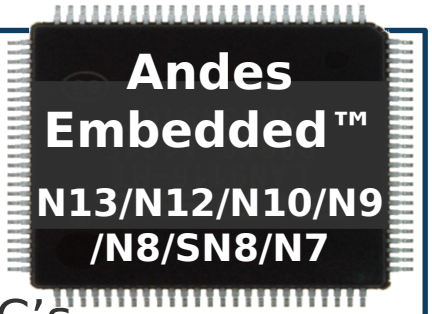
Corporate Highlights

- Founded in 2005 in Hsinchu Science Park, Taiwan, ROC
- Core team from AMD, DEC, Intel, MIPS, nVidia and Sun.
- Listed in EETimes' Silicon 60 Hot Startups to Watch (2012)

Technology and Products

- Innovate performance efficient processor IP solutions for SoC

Market and Business Status



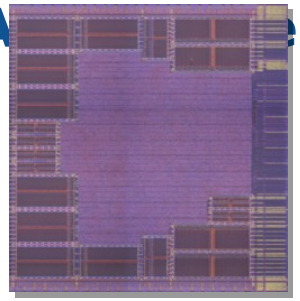
- Emerging applications
  - Smart and Green electronic devices
  - From Internet of Things to Cloud Computing
- >50 licensees and >300M Andes-Embedded™ SoC's
- >60 partners

Products and Infrastructure

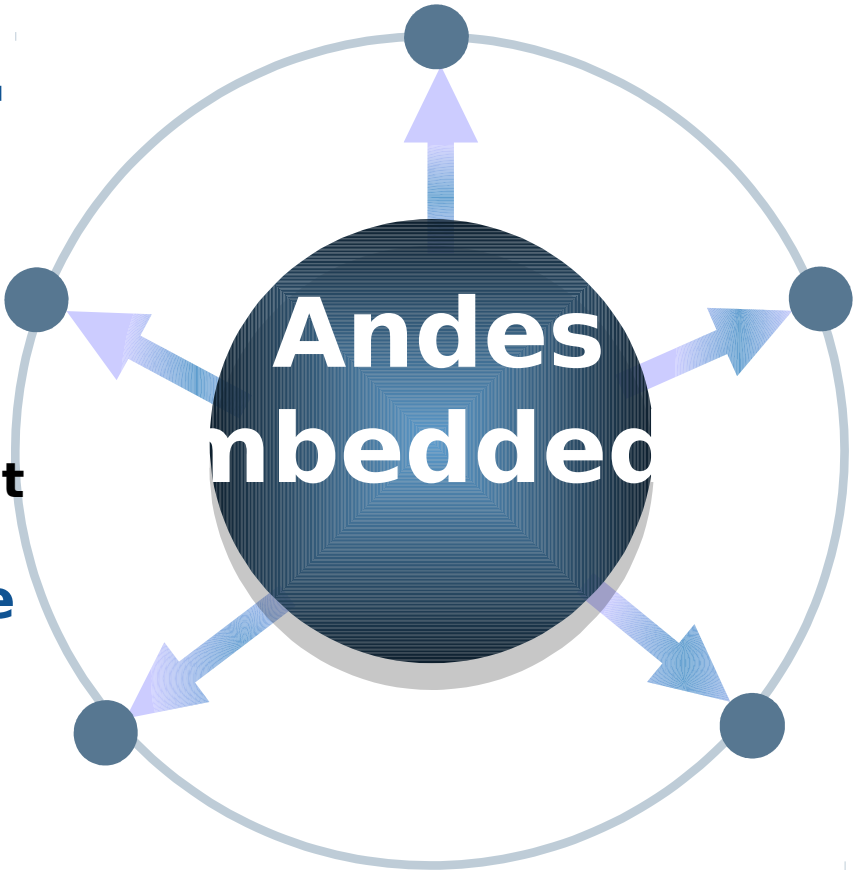
Processor Architecture
AndeStar™

```
smw.adm  $r1,[$sp],$r5,0x0
smw.adm  $sp,[$sp],$sp,0x2
addi     $sp,$sp,-8
sethi    $r1,0x50a
lwi      $r1,[$r1+#0x98]
mov55    $r2,$r0
mov55    $r0,$r1
lwi      $r1,[$r1+#0x8]
addi     $r3,$sp,12
```

Processor Cores

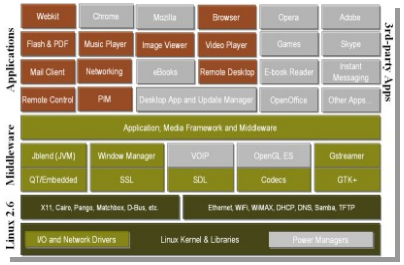


Development Tools
AndeSight™

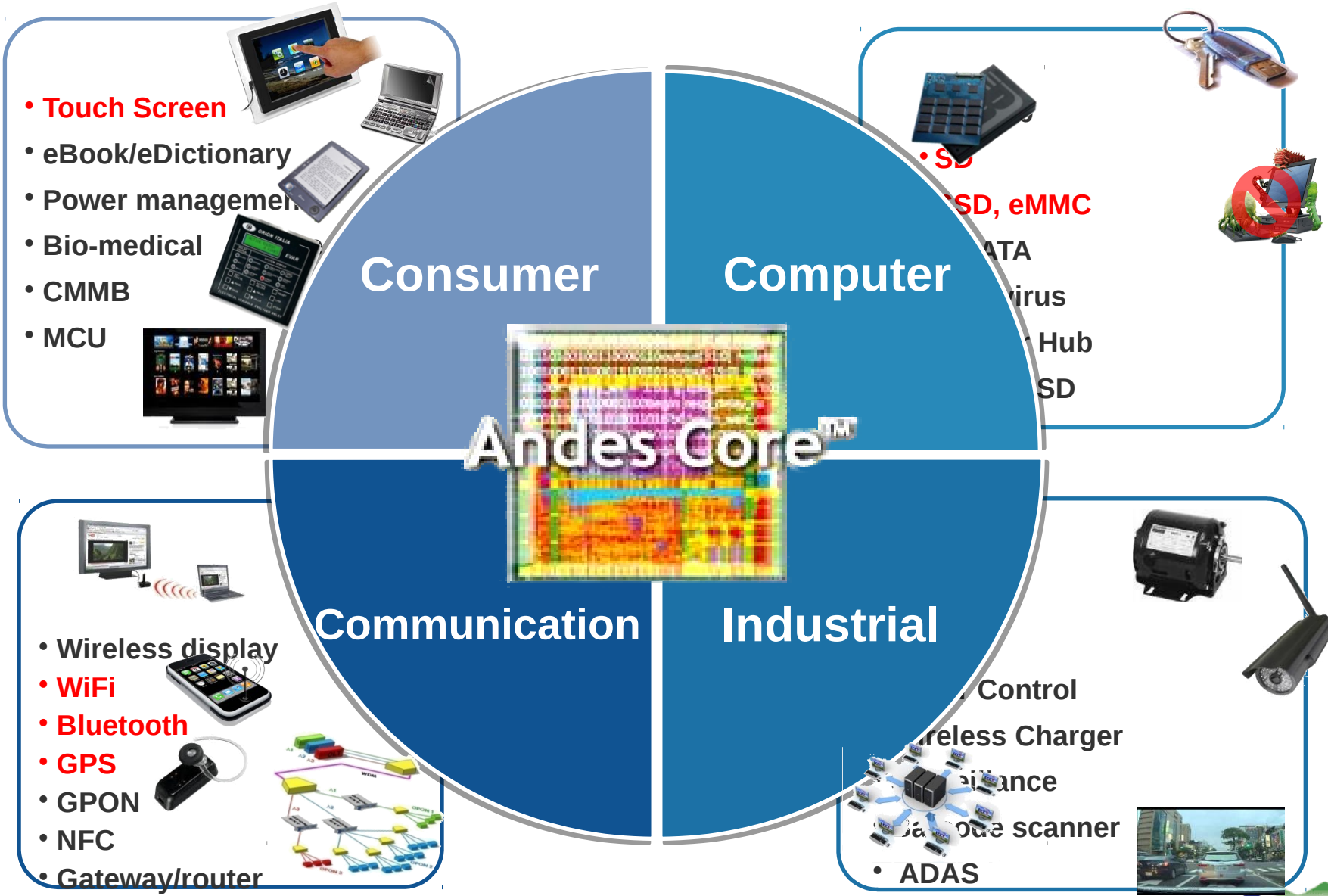


Development Platforms
AndeShape

Software Stacks
AndeSoft™



Examples of Andes-Embedded™ SoC



GCC Internal terms

- Machine Description
- RTL / RTX
- naming patterns
- predicate functions
- constraint / alternative
- Target Macros / Target Hooks
- Makefile fragment

A new target for GCC includes…

- Five necessary parts:
  1. configure for nds32 port
  2. nds32 machine description
  3. nds32 libgcc
  4. nds32 testsuite
  5. nds32 documentation

Add nds32 in configure part

1. configure

- 目標：
  - config.sub
  - gcc/config.gcc
  - libgcc/config.host

config.sub

```
293         | mipsr5900 | mipsr5900el \
294         | mipstx39 | mipstx39el \
295         | mn10200 | mn10300 \
296         | moxie \
297         | mt \
298         | msp430 \
299         | nds32 | nds32le | nds32be \
300         | nios | nios2 \
301         | ns16k | ns32k \
302         | open8 \
303         | or32 \
304         | pdp10 | pdp11 | pj | pjl \
```

gcc/config.gcc

```
2091 msp430*-*-*)
2092         tm_file="dbxelf.h elfos.h newlib-stdint.h ${tm_file}"
2093         c_target_objs="msp430-c.o"
2094         cxx_target_objs="msp430-c.o"
2095         target_has_targetm_common=no
2096         tmake_file="${tmake_file} msp430/t-msp430"
2097         ;;
2098 nds32le-*-*)
2099         target_cpu_default="0"
2100         tm_defines="${tm_defines}"
2101         tm_file="dbxelf.h elfos.h newlib-stdint.h ${tm_file}"
2102         tmake_file="nds32/t-mlibs"
2103         ;;
2104 nds32be-*-*)
2105         target_cpu_default="0|MASK_BIG_ENDIAN"
2106         tm_defines="${tm_defines} TARGET_BIG_ENDIAN_DEFAULT=1"
2107         tm_file="dbxelf.h elfos.h newlib-stdint.h ${tm_file}"
2108         tmake_file="nds32/t-mlibs"
2109         ;;
2110 pdp11-*-*)
2111         tm_file="${tm_file} newlib-stdint.h"
2112         use_gcc_stdint=wrap
2113         ;;
```

libgcc/config.host

```
853 msp430*-*-elf)
854         tmake_file="$tm_file t-crtstuff t-fdpbit msp430/t-msp430"
855         ;;
856 nds32*-elf*)
857         # Basic makefile fragment and extra_parts for crt stuff.
858         # We also append c-isr library implementation.
859         tmake_file="${tmake_file} nds32/t-nds32 nds32/t-nds32-isr"
860         extra_parts="crtbegin1.o crtend1.o libnds32_isr.a"
861         # Append library definition makefile fragment according to --wit
862         case "${with_nds32_lib}" in
863         "" | newlib)
864                 # Append library definition makefile fragment t-nds32-ne
865                 # Append 'soft-fp' software floating point make rule fra
866                 tmake_file="${tmake_file} nds32/t-nds32-newlib t-softfp-
867                 ;;
868         mculib)
869                 # Append library definition makefile fragment t-nds32-mc
870                 # The software floating point library is included in mcu
871                 tmake_file="${tmake_file} nds32/t-nds32-mculib"
872                 ;;
873         *)
874                 echo "Cannot accept --with-nds32-lib=$with_nds32_lib, av
875                 exit 1
876                 ;;
877         esac
878         ;;
879 pdp11-*-*)
880         tmake_file="pdp11/t-pdp11 t-fdpbit"
881         ;;
```

Add nds32 machine description

- 目標
  - gcc/common/config/nds32
  - gcc/config/nds32

```
30344
30345   return false;
30346
30347 }
30348
30349 /* Implement the TARGET_ASAN_SHADOW_OFFSET hook.  */
30350
30351 static unsigned HOST_WIDE_INT
30352 arm_asan_shadow_offset (void)
30353 {
30354   return (unsigned HOST_WIDE_INT) 1 << 29;
30355 }
30356
30357 #include "gt-arm.h"
"arm.c" 30357L, 891732C
```

- 如何新增一個自己的 target ？
  - 挑一個簡單的 target
    - 何謂簡單？行數少！
    - cp -a msp430 nds32
  - 回頭處理 gcc/config.gcc libgcc/config.host
    - 從 msp430 照抄，改名
    - 所有 msp430 → nds32

```
44070 #undef TARGET_FLOAT_EXCEPTIONS_ROUNDING_SUPPORTED_P
44071 #define TARGET_FLOAT_EXCEPTIONS_ROUNDING_SUPPORTED_P \
44072   ix86_float_exceptions_rounding_supported_p
44073
44074 struct gcc_target targetm = TARGET_INITIALIZER;
44075 ^L
44076 #include "gt-i386.h"
"i386.c" 44076L, 1386487C
```

```
2151   /* First we logically shift right by one.  Now we know
2152      that the top bit is zero and we can use the arithmetic
2153      right shift instruction to perform the rest of the shift.  */
2154   return "rrum.w\t#1, %0 ( rpt\t%Z2 ( rrax.w\t%0"; /* Six bytes.  */
2155 }
2156 ^L
2157 struct gcc_target targetm = TARGET_INITIALIZER;
2158
2159 #include "gt-msp430.h"
"msp430.c" 2159L, 58744C
```

Add nds32 machine description

- 目錄下的檔案改名以 nds32 開頭

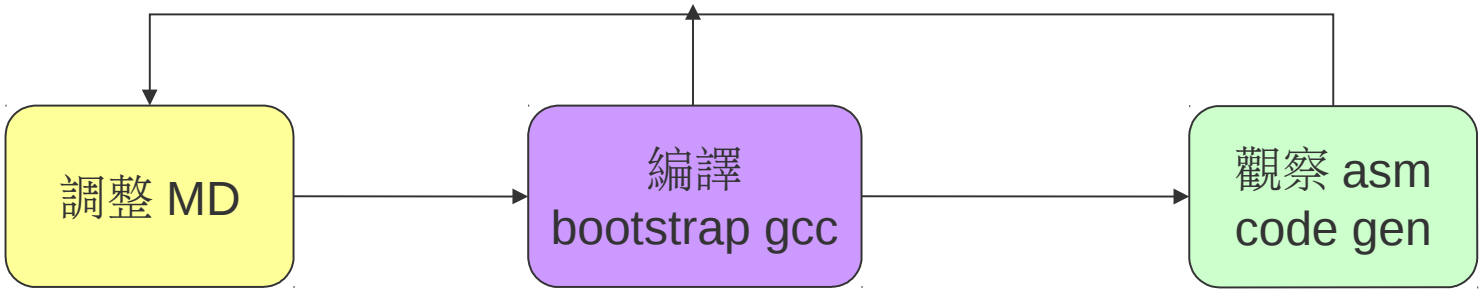
```
jasonwucj@atcsqa09 /playground/jasonwucj/try-hellogcc/gcc-4.9.0/gcc/config/nds32
$ ls
constraints.md  msp430.c  msp430-c.c  msp430.h  msp430.md  msp430-modes.def  msp430.opt  msp430-protos.h  predicates.md  README.txt  t-msp430
```

```
jasonwucj@atcsqa09 /playground/jasonwucj/try-hellogcc/gcc-4.9.0/gcc/config/nds32
$ ls
constraints.md  nds32.c  nds32-c.c  nds32.h  nds32.md  nds32-modes.def  nds32.opt  nds32-protos.h  predicates.md  README.txt  t-nds32
```

- 簡述相關檔案功能
  - nds32.c / nds32.h / nds32.md / nds32-protos.h
  - nds32.opt / nds32-opts.h
  - predicates.md / constraints.md
  - t-nds32

修改 back end 流程

- 所有 msp430_xxx 的 function 或 variables
  - 改名成 nds32_xxx



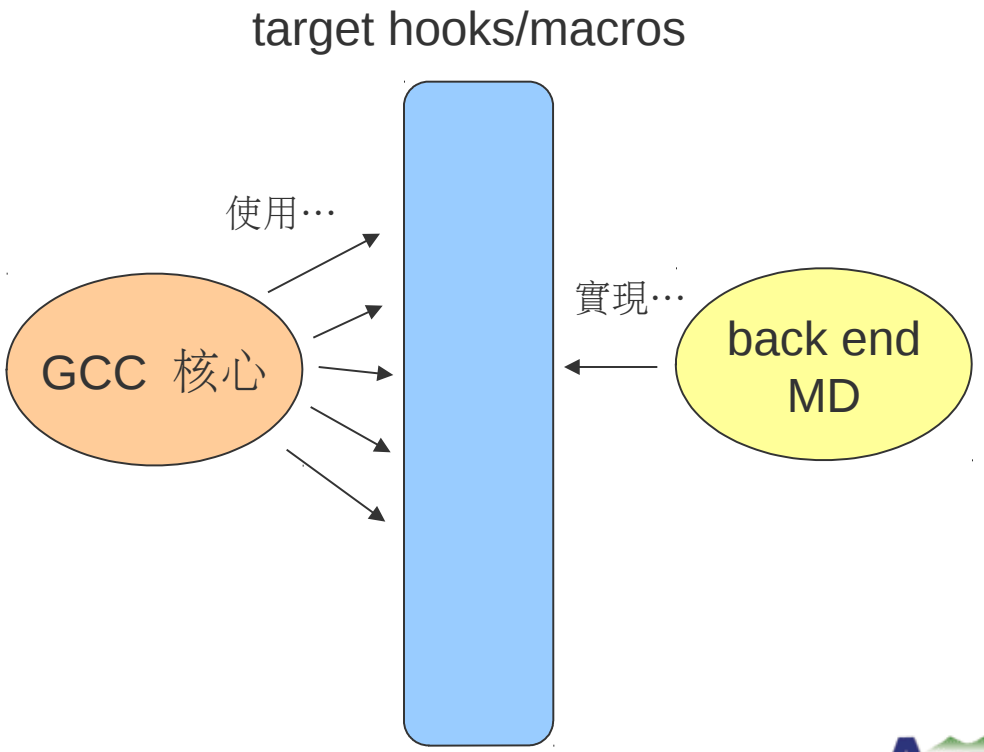

```
$ /source/gcc-4.9.0/configure \
--target=nds32le-elf \
--with-arch=v3 \
--prefix=…
… …
$ make all-gcc
$ make install-gcc
```

```
$ nds32le-elf-gcc -S test.c
```

開始調校 machine description

- Start to re-implement target specific parts
  - GCC Internal: Chapter.10
  - GCC Internal: Chapter.16 & Chapter.17

- 修改建議順序
  - nds32.h
  - nds32.md
  - nds32.c



nds32 registers

- nds32.h
  - 定義 data type size
  - 定義 target registers

```
825 #define REGISTER_NAMES                                          \
826 (                                                               \
827   "$r0",  "$r1",  "$r2",  "$r3",  "$r4",  "$r5",  "$r6",  "$r7",  \
828   "$r8",  "$r9",  "$r10", "$r11", "$r12", "$r13", "$r14", "$ta",  \
829   "$r16", "$r17", "$r18", "$r19", "$r20", "$r21", "$r22", "$r23", \
830   "$r24", "$r25", "$r26", "$r27", "$fp",  "$gp",  "$lp",  "$sp",  \
831   "$AP",                                                        \
832   "$SFP"                                                        \
833 }
```

```
451 /* Layout of Source Language Data Types.  */
452
453 #define INT_TYPE_SIZE           32
454 #define SHORT_TYPE_SIZE         16
455 #define LONG_TYPE_SIZE          32
456 #define LONG_LONG_TYPE_SIZE     64
457
458 #define FLOAT_TYPE_SIZE         32
459 #define DOUBLE_TYPE_SIZE        64
460 #define LONG_DOUBLE_TYPE_SIZE   64
461
462 #define DEFAULT_SIGNED_CHAR 1
463
464 #define SIZE_TYPE "long unsigned int"
465 #define PTRDIFF_TYPE "long int"
466 #define WCHAR_TYPE "short unsigned int"
467 #define WCHAR_TYPE_SIZE 16
```

```
508 /* Identifies the registers that are not available for
509    general allocation of values that must live across
510    function calls -- so they are caller-save registers.
511
512    0 : callee-save registers
513    1 : caller-save registers */
514 #define CALL_USED_REGISTERS             \
515 { /* r0  r1  r2  r3  r4  r5  r6  r7  */ \
516       1,  1,  1,  1,  1,  1,  0,  0,    \
517   /* r8  r9  r10 r11 r12 r13 r14 r15 */ \
518       0,  0,  0,  0,  0,  0,  0,  1,    \
519   /* r16 r17 r18 r19 r20 r21 r22 r23 */ \
520       1,  1,  1,  1,  1,  1,  1,  1,    \
521   /* r24 r25 r26 r27 r28 r29 r30 r31 */ \
522       1,  1,  1,  1,  0,  1,  0,  1,    \
523   /* ARG_POINTER:32 */                  \
524       1,                                \
525   /* FRAME_POINTER:33 */                \
526       1                                 \
527 }
```

nds32 registers

- nds32.h
  - 32-bit / 16-bit 混合式 instructions
  - 依據 register numbers 分等級

```
567 /* Register Classes.  */
568
569 /* In nds32 target, we have three levels of registers:
570      Low cost regsiters    : $r0 ~ $r7
571      Middle cost registers : $r8 ~ $r11, $r16 ~ $r19
572      High cost registers   : $r12 ~ $r14, $r20 ~ $r31
573
574    In practice, we have MIDDLE_REGS cover LOW_REGS regi
575    so that it provides more chance to use low cost regi
576 enum reg_class
577 {
578   NO_REGS,
579   R15_TA_REG,
580   STACK_REG,
581   LOW_REGS,
582   MIDDLE_REGS,
583   HIGH_REGS,
584   GENERAL_REGS,
585   FRAME_REGS,
586   ALL_REGS,
587   LIM_REG_CLASSES
588 };
```

```
592 #define REG_CLASS_NAMES \
593 {                       \
594   "NO_REGS",            \
595   "R15_TA_REG",         \
596   "STACK_REG",          \
597   "LOW_REGS",           \
598   "MIDDLE_REGS",        \
599   "HIGH_REGS",          \
600   "GENERAL_REGS",       \
601   "FRAME_REGS",         \
602   "ALL_REGS"            \
603 }
```

```
605 #define REG_CLASS_CONTENTS \
606 {                                                          \
607   {0x00000000, 0x00000000}, /* NO_REGS      :            */ \
608   {0x00008000, 0x00000000}, /* R15_TA_REG   : 15         */ \
609   {0x80000000, 0x00000000}, /* STACK_REG    : 31         */ \
610   {0x000000ff, 0x00000000}, /* LOW_REGS     : 0-7        */ \
611   {0x000f0fff, 0x00000000}, /* MIDDLE_REGS  : 0-11, 16-19*/ \
612   {0xfff07000, 0x00000000}, /* HIGH_REGS    : 12-14, 20-31*/ \
613   {0xffffffff, 0x00000000}, /* GENERAL_REGS: 0-31        */ \
614   {0x00000000, 0x00000003}, /* FRAME_REGS   : 32, 33     */ \
615   {0xffffffff, 0x00000003}  /* ALL_REGS     : 0-31, 32, 33*/ \
616 }
617
```

nds32 spec 設定

- nds32.h
  - Control compilation driver
  - http://gcc.gnu.org/onlinedocs/gcc/Spec-Files.html

```
327 #define OPTION_DEFAULT_SPECS \
328   {"arch", "%{!march=*:-march=%(VALUE)}" }
329
330 #define CC1_SPEC \
331   ""
332
333 #define ASM_SPEC \
334   " %{mbig-endian:-EB} %{mlittle-endian:-EL}"
335
336 /* If user issues -mrelax, -mforce-fp-as-gp, or -mex9,
337    we need to pass '--relax' to linker.
338    Besides, for -mex9, we need to further pass '--mex9'.  */
339 #define LINK_SPEC \
340   " %{mbig-endian:-EB} %{mlittle-endian:-EL}" \
341   " %{mrelax|mforce-fp-as-gp|mex9:--relax}" \
342   " %{mex9:--mex9}"
343
344 #define LIB_SPEC \
345   " -lc -lgloss"
```

如果有 -mbig-endian,
就傳給 assembler -EB

如果有
-mrelax 或 -mforce-fp-as-gp
或 -mex9，就傳給 linker --relax

直接使用 -lc -lgloss
來指示要一起 link 的 library

```
361 #define STARTFILE_SPEC \
362   " %{!mno-ctor-dtor:crt1.o%s;:crt0.o%s}" \
363   " %{!mno-ctor-dtor:crtbegin1.o%s}"
364 #define ENDFILE_SPEC \
365   " %{!mno-ctor-dtor:crtend1.o%s}"
```

如果沒有 -mno-ctor-dtor,
就拉進 crtend1.o

Fundamental naming patterns

- nds32.md
  - move patterns
    - movqi, movhi, movsi
  - boolean operations
    - andsi3, iorsi3, xorsi3
  - add/sub/mul/div/shift
    - addsi3, subsi3, mulsi3, divmodsi3, udivmodsi3, ashlsi3, ashrsi3, lshrsi3
  - compare and branch operations
    - cbranchsi4
  - prologue and epilogue
    - prologue, epilogue

nds32 move pattern

- nds32.md - move pattern

```
135 (define_insn "*mov<mode>"
136   [(set (match_operand:QIHISI 0 "nonimmediate_operand" "=r, r, U45, U33, U37, U45, m,   l,   l,   l,   d, r,    d,    r,    r,    r")
137         (match_operand:QIHISI 1 "nds32_move_operand"   " r, r,   l,   l,   l,   d, r, U45, U33, U37, U45, m, Ip05, Is05, Is20, Ihig"))]
138   ""
139 {
140   switch (which_alternative)
141     {
142     case 0:
143       return "mov55\t%0, %1";
144     case 1:
145       return "ori\t%0, %1, 0";
146     case 2:
147     case 3:
148     case 4:
149     case 5:
150       return nds32_output_16bit_store (operands, <byte>);
151     case 6:
152       return nds32_output_32bit_store (operands, <byte>);
153     case 7:
154     case 8:
155     case 9:
156     case 10:
157       return nds32_output_16bit_load (operands, <byte>);
158     case 11:
159       return nds32_output_32bit_load (operands, <byte>);
160     case 12:
161       return "movpi45\t%0, %1";
162     case 13:
163       return "movi55\t%0, %1";
164     case 14:
165       return "movi\t%0, %1";
166     case 15:
167       return "sethi\t%0, hi20(%1)";
168     default:
169       gcc_unreachable ();
170     }
171 }
172   [(set_attr "type"   "alu,alu,store,store,store,store,store,load,load,load,load,load,alu,alu,alu,alu")
173    (set_attr "length" "  2,  4,    2,    2,    2,    2,    4,   2,   2,   2,   2,   4,  2,  2,  4,  4")])
```

利用 predicate 來限制指令的可能性

搭配不同的 constraints/alternative

不同的指令

可設定此指令的 type 和 length

nds32 boolean pattern

- nds32.md - boolean pattern

```
491 (define_insn "andsi3"
492   [(set (match_operand:SI 0 "register_operand"          "= w, r,    l,    l,    l,    l,
493         (and:SI (match_operand:SI 1 "register_operand" " %0, r,    l,    l,    l,    l,
494                 (match_operand:SI 2 "general_operand"  "  w, r, Izeb, Izeh, Ixls, Ix11,
495   ""
496 {
497   HOST_WIDE_INT mask = INTVAL (operands[2]);
498   int zero_position;
499
500   /* 16-bit andi instructions:
501      andi Rt3,Ra3,0xff    -> zeb33  Rt3,Ra3
502      andi Rt3,Ra3,0xffff -> zeh33  Rt3,Ra3
503      andi Rt3,Ra3,0x01    -> xlsb33 Rt3,Ra3
504      andi Rt3,Ra3,0x7ff   -> x11b33 Rt3,Ra3
505      andi Rt3,Rt3,2^imm3u          -> bmski33 Rt3,imm3u
506      andi Rt3,Rt3,(2^(imm3u+1))-1  -> fexti33 Rt3,imm3u.   */
507
508   switch (which_alternative)
509     {
510     case 0:
511       return "and33\t%0, %2";
512     case 1:
513       return "and\t%0, %1, %2";
514     case 2:
515       return "zeb33\t%0, %1";
516     case 3:
517       return "zeh33\t%0, %1";
518     case 4:
519       return "xlsb33\t%0, %1";
520     case 5:
521       return "x11b33\t%0, %1";
```

不同的 constraints

and 指令有各式各樣的變化

不同的 asm 指令

predicates and constraints

- predicates.md & constraints.md
  - GCC Internal: 16.8.7 Defining Machine-Specific Constraints

```
46 (define_predicate "nds32_move_operand"
47   (and (match_operand 0 "general_operand")
48        (not (match_code "high,const,symbol_ref,label_ref"))))
49 {
50   /* If the constant op does NOT satisfy Is20 nor Ihig,
51      we can not perform move behavior by a single instructi
52   if (CONST_INT_P (op)
53       && !satisfies_constraint_Is20 (op)
54       && !satisfies_constraint_Ihig (op))
55     return false;
56
57   return true;
58 })
59
60 (define_special_predicate "nds32_load_multiple_operation"
61   (match_code "parallel")
62 {
63   /* To verify 'load' operation, pass 'true' for the second
64      See the implementation in nds32.c for details.  */
65   return nds32_valid_multiple_load_store (op, true);
66 })
```

```
31 (define_register_constraint "l" "LOW_REGS"
32   "LOW register class $r0 ~ $r7")
33
34 (define_register_constraint "d" "MIDDLE_REGS"
35   "MIDDLE register class $r0 ~ $r11, $r16 ~ $r19")
36
37 (define_register_constraint "h" "HIGH_REGS"
38   "HIGH register class $r12 ~ $r14, $r20 ~ $r31")
39
40
41 (define_register_constraint "t" "R15_TA_REG"
42   "Temporary Assist register $ta (i.e. $r15)")
43
44 (define_register_constraint "k" "STACK_REG"
45   "Stack register $sp")
46
47
48 (define_constraint "Iu03"
49   "Unsigned immediate 3-bit value"
50   (and (match_code "const_int")
51        (match_test "ival < (1 << 3) && ival >= 0")))
52
```

若需要另外呼叫函式:
在 nds32-protos.h 定義 prototypes
在 nds32.c 實作

善用 match_code 和 match_test
可使用 satisfies_constraint_Iu03()

實作 target hooks

- nds32.c

```
5586 /* Describing Relative Costs of Operations.  */
5587
5588 #undef TARGET_REGISTER_MOVE_COST
5589 #define TARGET_REGISTER_MOVE_COST nds32_register_move_cost
5590
5591 #undef TARGET_MEMORY_MOVE_COST
5592 #define TARGET_MEMORY_MOVE_COST nds32_memory_move_cost
5593
5594 #undef TARGET_RTX_COSTS
5595 #define TARGET_RTX_COSTS nds32_rtx_costs
5596
5597 #undef TARGET_ADDRESS_COST
5598 #define TARGET_ADDRESS_COST nds32_address_cost
5599
```

1. 先 undef 掉預設的 target hook
2. 接上自己的 hook 實作

```
5673 /* Defining target-specific uses of __att
5674
5675 #undef TARGET_ATTRIBUTE_TABLE
5676 #define TARGET_ATTRIBUTE_TABLE nds32_attr
5677
5678 #undef TARGET_MERGE_DECL_ATTRIBUTES
5679 #define TARGET_MERGE_DECL_ATTRIBUTES nds3
5680
5681 #undef TARGET_INSERT_ATTRIBUTES
5682 #define TARGET_INSERT_ATTRIBUTES nds32_in
5683
5684 #undef TARGET_OPTION_PRAGMA_PARSE
5685 #define TARGET_OPTION_PRAGMA_PARSE nds32_option_pragma_parse
5686
5687 #undef TARGET_OPTION_OVERRIDE
5688 #define TARGET_OPTION_OVERRIDE nds32_option_override
5689
```

```
2414 /* Describing Relative Costs of Operations.  */
2415
2416 static int nds32_register_move_cost (enum machine_mode mode
2417                                      reg_class_t from,
2418                                      reg_class_t to)
2419 {
2420   if (from == HIGH_REGS || to == HIGH_REGS)
2421     return 6;
2422
2423   return 2;
2424 }
```

提供 target specific options

- Start to add target specific options
  - GCC Internal: Chapter.8

- 修改
  - nds32.opt
    - 設計可用的 options
  - nds32-opts.h
    - 輔助 nds32.opt 的 enum 定義

新增、調整、與控制 options

- nds32.opt

沒有 -mno-reduced-regs

提供 MASK_REDUCED_REGS
可用 TARGET_REDUCED_REGS 檢查
-mfull-regs 會清掉此 MASK 設定

```
32 mreduced-regs
33 Target Report RejectNegative Negative(mfull-regs) Mask(REDUCED_REGS)
34 Use reduced-set registers for register allocation.
35
36 mfull-regs
37 Target Report RejectNegative Negative(mreduced-regs) InverseMask(REDUCED_REGS)
38 Use full-set registers for register allocation.
39
40 mcmov
41 Target Report Mask(CMOV)
42 Generate conditional move instructions.
```

以 Negative 讓 -mfull-regs 和 -mreduced-regs 互斥

TARGET_CMOV 檢查
-mcmov / -mno-cmov

```
3252   /* See if we are using reduced-set registers:
3253        $r0~$r5, $r6~$r10, $r15, $r28, $r29, $r30, $r31
3254      If so, we must forbid using $r11~$r14, $r16~$r27.  */
3255   if (TARGET_REDUCED_REGS)
3256     {
3257       int r;
3258
3259       /* Prevent register allocator from
3260          choosing it as doing register allocation.  */
3261       for (r = 11; r <= 14; r++)
3262         fixed_regs[r] = call_used_regs[r] = 1;
3263       for (r = 16; r <= 27; r++)
3264         fixed_regs[r] = call_used_regs[r] = 1;
3265     }
```

新增、調整、與控制 options

- nds32.opt / nds32-opts.h

```
68 march=
69 Target RejectNegative Joined Enum(nds32_arch_type) Var(nds32_arch_option) Init(ARCH_V3)
70 Specify the name of the target architecture.
71
72 Enum
73 Name(nds32_arch_type) Type(enum nds32_arch_type)
74
75 EnumValue
76 Enum(nds32_arch_type) String(v2) Value(ARCH_V2)
77
78 EnumValue
79 Enum(nds32_arch_type) String(v3) Value(ARCH_V3)
80
81 EnumValue
82 Enum(nds32_arch_type) String(v3m) Value(ARCH_V3M)
```

```
27 /* The various ANDES ISA.  */
28 enum nds32_arch_type
29 {
30   ARCH_V2,
31   ARCH_V3,
32   ARCH_V3M
33 };
```

nds32-opts.h 中定義 enum
提供 -march=XXX 的設定

Refine machine description design

- How to use unnamed patterns
  - sample of add_slli / add_srli

```
318 ;; GCC intends to simplify (plus (ashift ...) (reg))
319 ;; into (plus (mult ...) (reg)), so our matching pattern takes 'mult'
320 ;; and needs to ensure it is exact_log2 value.
321 (define_insn "*add_slli"
322   [(set (match_operand:SI 0 "register_operand"                    "=r")
323         (plus:SI (mult:SI (match_operand:SI 1 "register_operand"   " r")
324                           (match_operand:SI 2 "immediate_operand" " i"))
325                  (match_operand:SI 3 "register_operand"           " r"))))]
326   "TARGET_ISA_V3
327    && (exact_log2 (INTVAL (operands[2])) != -1)
328    && (exact_log2 (INTVAL (operands[2])) <= 31)"
329 {
330   /* Get floor_log2 of the immediate value
331      so that we can generate 'add_slli' instruction.  */
332   operands[2] = GEN_INT (floor_log2 (INTVAL (operands[2])));
333
334   return "add_slli\t%0, %3, %1, %2";
335 }
336   [(set_attr "type" "alu")
337    (set_attr "length" "4")])
338
339 (define_insn "*add_srli"
340   [(set (match_operand:SI 0 "register_operand"                    "=    r")
341         (plus:SI (lshiftrt:SI (match_operand:SI 1 "register_operand"  "    r")
342                               (match_operand:SI 2 "immediate_operand" " Iu05"))
343                  (match_operand:SI 3 "register_operand"           "    r"))))]
344   "TARGET_ISA_V3"
345   "add_srli\t%0, %3, %1, %2"
346   [(set_attr "type" "alu")
347    (set_attr "length" "4")])
```

設計 unnamed pattern 來提供 GCC 組合出不同 RTX 的機會

也是可以使用自定的 constraints

Refine machine description design

- How to enable LRA
  - A replacement of GCC reload phase
  - Use target hook: **TARGET_LRA_P**
  - Use reload-stuff as less as you can
  - DO NOT use **`reload_in_progress`**, use **`lra_in_progress`** instead
  - It is OK to use **`reload_complete`**
  - Note that the meaning of constraint modifier '*' is different between reload and LRA

Refine machine description design

❖ Sample of enabling LRA

```
5462 /* Register Classes.  */
5463
5464 #undef TARGET_CLASS_MAX_NREGS
5465 #define TARGET_CLASS_MAX_NREGS nds32_class_max_nregs
5466
5467 #undef TARGET_LRA_P
5468 #define TARGET_LRA_P hook_bool_void_true
5469
5470 #undef TARGET_REGISTER_PRIORITY
5471 #define TARGET_REGISTER_PRIORITY nds32_register_priority
5472
```

```
d"                              "=l, r,    l, *r")
1 "nonimmediate_operand" " l, r, U33,  m"))))]
```



```
2319     case SYMBOL_REF:
2320
2321       if (!TARGET_GP_DIRECT
2322           && (reload_completed
2323               || reload_in_progress
2324               || lra_in_progress))
2325         return false;
2326
2327       /* (mem (symbol_ref A)) => [symbol_ref] */
2328       return !currently_expanding_to_rtl;
2329
2330     case CONST:
2331
2332       if (!TARGET_GP_DIRECT
2333           && (reload_completed
2334               || reload_in_progress
2335               || lra_in_progress))
2336         return false;
```

注意 ‘*’ modifier 在 LRA 與 reload 的意義不一樣

如果你要讓你的 target 同時存在著 reload 與 LRA 的機會，在設計 MD patterns 時要非常小心

GCC Internal: Chapter 16.8.4

Add nds32 libgcc

- 目標
  - libgcc/config/nds32

- libgcc 與 standard library 不同
  - 提供 implicitly function call 的需求
  - long long a, b, c;
    c = a + b;
    - call __adddi3
  - float x, y, z;
    z = x * y;
    - call __mulsf3

makefile fragment

- t-xxx 檔案
- 參考 libgcc/Makefile.in 行爲

```
340
341 tmake_file = @tmake_file@
342 include $(srcdir)/empty.mk $(tmake_file)
343
```

- GCC Internal: Chapter.19

- libgcc/config/ 下有許多 makefile fragment
- 也可提供 nds32 專屬的 makefile fragment

nds32 libgcc settings

- 由 libgcc/config.host 設定所有必要資訊

```
856 nds32*-elf*)
857         # Basic makefile fragment and extra_parts for crt stuff.
858         # We also append c-isr library implementation.
859         tmake_file="${tmake_file} nds32/t-nds32 nds32/t-nds32-isr"
860         extra_parts="crtbegin1.o crtend1.o libnds32_isr.a"
861         # Append library definition makefile fragment according to --with-nds32-lib=X setting.
862         case "${with_nds32_lib}" in
863         "" | newlib)
864                 # Append library definition makefile fragment t-nds32-newlib.
865                 # Append 'soft-fp' software floating point make rule fragment provided by gcc.
866                 tmake_file="${tmake_file} nds32/t-nds32-newlib t-softfp-sfdf t-softfp"
867                 ;;
868         mculib)
869                 # Append library definition makefile fragment t-nds32-mculib.
870                 # The software floating point library is included in mculib.
871                 tmake_file="${tmake_file} nds32/t-nds32-mculib"
872                 ;;
873         *)
874                 echo "Cannot accept --with-nds32-lib=$with_nds32_lib, available values are: newlib mculib" 1>&2
875                 exit 1
876                 ;;
877         esac
878         ;;
```

Add nds32 testsuite

- 目標
  - gcc/testsuite
  - gcc/testsuite/gcc.target/nds32
- GCC testsuite is using DejaGNU framework
  - GCC Internal: Chapter.7
  - http://gcc.gnu.org/install/test.html
- gcc/testsuite/lib/target-supports.exp
  - 設計屬於自己 target 的檢查

```
459
460 # Return true if profiling is supported on the target.
461
462 proc check_profiling_available { test_what } {
463     global profiling_available_saved
```

```
528             || [istarget mips*-*-elf*]
529             || [istarget mmix-*-*]
530             || [istarget mn10300-*-elf*]
531             || [istarget moxie-*-elf*]
532             || [istarget msp430-*-*]
533             || [istarget nds32*-*-elf]
534             || [istarget picochip-*-*]
```

general gcc testcases

- gcc/testsuite/*

```
87 --- gcc/testsuite/gcc.dg/stack-usage-1.c
88 +++ gcc/testsuite/gcc.dg/stack-usage-1.c
89 @@ -38,6 +38,9 @@
90  #  else
91  #    define SIZE 248
92  #  endif
93 +#elif defined (__nds32__)
94 +#  define SIZE 248 /* 256 - 8 bytes, only $fp and padding bytes are saved in
95 +                      the register save area under O0 optimization level.  */
96  #elif defined (__powerpc64__) || defined (__ppc64__) || defined (__POWERPC64__) \
97        || defined (__PPC64__)
```

讓結果可以 PASS

```
101 --- gcc/testsuite/gcc.dg/torture/pr37868.c
102 +++ gcc/testsuite/gcc.dg/torture/pr37868.c
103 @@ -1,6 +1,6 @@
104  /* { dg-do run } */
105  /* { dg-options "-fno-strict-aliasing" } */
106 -/* { dg-skip-if "unaligned access" { arc*-*-* epiphany-*-* sparc*-*-* sh*-*-* tic6x-*-* } "*" "" } */
107 +/* { dg-skip-if "unaligned access" { arc*-*-* epiphany-*-* nds32*-*-* sparc*-*-* sh*-*-* tic6x-*-* } "*" "" } */
```

令此 testcase 編列為 UNSUPPORTED

```
61 --- gcc/testsuite/gcc.dg/sibcall-3.c
62 +++ gcc/testsuite/gcc.dg/sibcall-3.c
63 @@ -5,7 +5,7 @@
64     Copyright (C) 2002 Free Software Foundation Inc.
65     Contributed by Hans-Peter Nilsson  <hp@bitrange.com>  */
66
67 -/* { dg-do run { xfail { { cris-*-* crisv32-*-* h8300-*-* hppa*64*-*-* m32r-*-* mcore-*-* mn10300-*-* xstormy16-*-* v850*-*-
   32 } } } } } */
68 +/* { dg-do run { xfail { { cris-*-* crisv32-*-* h8300-*-* hppa*64*-*-* m32r-*-* mcore-*-* mn10300-*-* nds32*-*-* xstormy16-*
    && { ! arm32 } } } } } */
69  /* -mlongcall disables sibcall patterns.  */
70  /* { dg-skip-if "" { powerpc*-*-* } { "-mlongcall" } { "" } } */
71  /* { dg-options "-O2 -foptimize-sibling-calls" } */
```

令此 testcase 編列為 XFAIL

nds32 specific testcases

- gcc/testsuite/gcc.target/nds32

```
jasonwucj@atcsqa09 /playground/jasonwucj/try-hellogcc/gcc-4.9.0/gcc/testsuite/gcc.target/nds32
$ ls
basic-main.c  builtin-isb.c  builtin-isync.c  builtin-mfsr-mtsr.c  builtin-mfusr-mtusr.c  builtin-setgie-dis.c  builtin-setgie-en.c  nds32.exp
```

```
172 +++ gcc/testsuite/gcc.target/nds32/basic-main.c
173 @@ -0,0 +1,9 @@
174 +/* This is a basic main function test program.  */
175 +
176 +/* { dg-do run { target nds32*-*-* } }  */
177 +/* { dg-options "-O0" }  */
178 +
179 +int main(void)
180 +{
181 +  return 0;
182 +}
```

```
187 +++ gcc/testsuite/gcc.target/nds32/builtin-isb.c
188 @@ -0,0 +1,11 @@
189 +/* Verify that we generate isb instruction with
190 +
191 +/* { dg-do compile { target nds32*-*-* } }  */
192 +/* { dg-options "-O0" }  */
193 +/* { dg-final { scan-assembler "\\tisb" } }  */
194 +
195 +void
196 +test (void)
197 +{
198 +  __builtin_nds32_isb ();
199 +}
```

```
224 +/* Verify that we generate mfsr/mtsr instruction with
225 +
226 +/* { dg-do compile { target nds32*-*-* } }  */
227 +/* { dg-options "-O0" }  */
228 +/* { dg-final { scan-assembler "\\tmfsr" } }  */
229 +/* { dg-final { scan-assembler "\\tmtsr" } }  */
230 +
231 +#include <nds32_intrinsic.h>
232 +
233 +void
234 +test (void)
235 +{
236 +  int ipsw_value;
237 +
238 +  ipsw_value = __builtin_nds32_mfsr (__NDS32_REG_IPSW__);
239 +  __builtin_nds32_mtsr (ipsw_value, __NDS32_REG_IPSW__);
240 +}
```

善用:

- dg-do
- dg-options
- dg-final

Add nds32 documentation

- 目標
  - gcc/doc/extend.texi
  - gcc/doc/install.texi
  - gcc/doc/invoke.texi
  - gcc/doc/md.texi

- 影響　documentation
  - GCC User Manual
  - GCC Internal

Add nds32 documentation

- gcc/doc/extend.texi
  - 描述 nds32 專屬 attritube
  - 描述 nds32 專屬 builtin function
- gcc/doc/install.texi
  - 描述 nds32 專屬的 configure 階段 options
- gcc/doc/invoke.texi
  - 描述 nds32 gcc 專屬的 options
- gcc/doc/md.texi
  - 描述 GCC Internal 文件中對 nds32 專屬的 porting

Add nds32 documentation

- Tex/LaTeX 語法
  - make pdf / make html

```
161  @emph{MSP430 Options}
162  @gccoptlist{-msim -masm-hex -mmcu= -mlarge -msmall -mrelax}
163
164 +@emph{NDS32 Options}
165 +@gccoptlist{-mbig-endian -mlittle-endian @gol
166 +-mreduced-regs -mfull-regs @gol
167 +-mcmov -mno-cmov @gol
168 +-mperf-ext -mno-perf-ext @gol
169 +-mv3push -mno-v3push @gol
170 +-m16bit -mno-16bit @gol
171 +-mgp-direct -mno-gp-direct @gol
172 +-misr-vector-size=@var{num} @gol
173 +-mcache-block-size=@var{num} @gol
174 +-march=@var{arch} @gol
175 +-mforce-fp-as-gp -mforbid-fp-as-gp @gc
176 +-mex9 -mctor-dtor -mrelax}
177 +
178  @emph{PDP-11 Options}
179  @gccoptlist{-mfpu  -msoft-float  -mac(
180  -mbcopy  -mbcopy-builtin  -mint32  -mi
```

```
 96  once the handler returns.
 97  @end table
 98
 99 +@node NDS32 Built-in Functions
100 +@subsection NDS32 Built-in Functions
101 +
102 +These built-in functions are available for the NDS32 target:
103 +
104 +@deftypefn {Built-in Function} void __builtin_nds32_isync (int *@var{addr})
105 +Insert an ISYNC instruction into the instruction stream where
106 +@var{addr} is an instruction address for serialization.
107 +@end deftypefn
108 +
109 +@deftypefn {Built-in Function} void __builtin_nds32_isb (void)
110 +Insert an ISB instruction into the instruction stream.
111 +@end deftypefn
112 +
113 +@deftypefn {Built-in Function} int __builtin_nds32_mfsr (int @var{sr})
114 +Return the content of a system register which is mapped by @var{sr}.
115 +@end deftypefn
116 +
```

Summary

1. configure for nds32 port
   - config.sub
   - gcc/config.gcc
   - libgcc/config.host
2. nds32 machine description
   - gcc/config/nds32
   - gcc/common/config/nds32
3. nds32 libgcc
   - libgcc/config/nds32
4. nds32 testsuite
   - gcc/testsuite
   - gcc/testsuite/gcc.target/nds32
5. nds32 documentation
   - gcc/doc

Q & A time

Thank You !!