Dokumentacja przeniesienia oprogramowania ANSA 4.0 na komputery RS6000

Andrzej Król

Instytut Informatyki
AGH

Marzec 1993

Spis treści

1 **2**

1.1 Wprowadzenie 2

1.2 Przygotowanie środowiska pod system operacyjny AIX 2

2 **4**

2.1 Nowe zbiory konfiguracyjne 4

2.2 Uzupełnienia w istniejących zbiorach konfiguracyjnych 9

2.3 Poprawki w zbiorach spowodowane zmianą środowiska 16

3 **19**

3.1 Generacja systemu ANSA 19

Rozdział 1

1.1 Wprowadzenie

Poniższe opracowanie stanowi podsumowanie prac związanych z przeniesieniem oprogramowania ANSA na komputery typu RS6000 pracujące pod kontrolą systemu operacyjnego AIX 3.1.5. Przy instalacji oprogramowania wykorzystano wskazówki zawarte w rozdziale 11 dokumenatcji ANSAware 4.0 System Manager's Guide. Oprócz normalnych trudności zawiązanych z utworzeniem odpowiednich zbiorów konfiguracyjnych dla danej platformy sprzętowej pojawiły się nowe, związane z charakterystycznymi właściwościami systemu operacyjnego AIX 3.1.5. Odnośne zbiory konfiguracyjne i poprawki do zbiorów źródłowych systemu ANSA przedstawione w niniejszym opracowaniu, stanowią kompletną dokumentację budowy platformy ANSA dla maszyn typu RS6000. Zostały one przedstawione w rozdzaile 2. Zagadnienie generacji oprogramowania ANSA dla platformy RS6000 stanowi przedmiot rozdziału 3.

1.2 Przygotowanie środowiska pod system operacyjny AIX

System operacyjny AIX jest zasadniczo systemem typu SYSV, jednak starano się w nim zaimplementować maksymalnie dużo rozwiązań z systemu typu BSD. Tworząc nową paltformę dla oprogramowania ANSA wybrano drogę maksynalnej zgodności z systemem BSD. Wymagało to przygotowania środowiska kompilatora C. Osobny problem stanowi preprocesor cpp dostraczany standardowo z sytemem AIX. Preprocesor /lib/cpp nie jest zgodny ze specyfikacją preprocesora cpp Kernighan & Ritche. Użycie jego wymagałoby gruntownych przeróbek i porawek w zbiorach Imake co jest zadaniem trudnym stwarzającym duże możliwości pomyłek. Na szczęście istnieje możliwość użycia preprocesora dostarczanego razem z oprogramowaniem X11 (preprocesor /usr/lpp/X11/Xamples/util/cpp/cpp). Zapewnia on zgodność ze standardem K&R.

Aby móc skorzystać ze zgodności ze systemem BSD niezbędne jest odpowiednie przygotowanie systemowego kompilatora C. W tym celu do zbioru /etc/xlc.cfg dodano definicję środowiska dla kompilatora C zapewniającą kompatybilność z systemem BSD:

```
* standard bsdc compiler aliased as bsdcc
bsdcc:  use        = DEFLT
        crt        = /lib/crt0.o
        mcrt       = /lib/mcrt0.o
        gcrt       = /lib/gcrt0.o
        libraries  = -lbsd, -lc
        proflibs   = -L/lib/profiled,-L/usr/lib/profiled
        options    = -H512,-T512, -qlanglvl=extended, -qnoro, -D_BSD, -D_NONSTD_
TYPES, -D_NO_PROTO, -D_BSD_INCLUDES, -bnodelcsect, -U__STR__, -U__MATH__
```

Rozdział 2

W celu utworzenia nowej platformy dla oprogramowania ANSA konieczne jest przygotowanie odpowiednich zbiorów konfiguracyjnych. Należy też wykonać pewne poprawki w kodzie źródłowym.

2.1 Nowe zbiory konfiguracyjne

- aix.h
  Zbiór ten należy utworzyć w katalogu master/include/capsule/stack. Opisywana implementacja ANSY na komputerze RS6000 nie wykorzystuje standardowych procedur Unixa **setjmp** i **longjmp** do przełączania stosu. Powodem jest fakt sprawdzania przez procedurę **longjmp** czy powrót następuje do środowiska zachowanego przez procedurę **setjmp**

```
/* Header Start */
/*
 *  Copyright 1989 APM Limited (ISA Project)
 *
 *  File: aix.h
 *  Date Created:
 *
 *  General Description:
 *    Interface to stack handler for IBMs RS/6000 using our own stack handle
 *    function
 *
 *  $Header: /disc2/local/src/ANSAware3.0Dist/master/include/capsule/stack/RCS/u
nix.h,v 1.2 90/03/16 11:40:10 can Exp $
 *
 */
/* Header End */

#ifndef __STACK_INCLUDED__
#define __STACK_INCLUDED__

#include "options.h"
```

```
/*
 * This file sets up constants to allow multi-tasking.  This requires the
 * infrastructure associating a stack and register dump with each task -
 * actually a task IS the stack and register dump.
 *
 * We assume a mechanism like Unix setjmp/longjmp where an array (called
 * a jmp_buf) holds all volatile state (i.e. PSW and registers) associated
 * with an executing task.  Within this array, two fields describing the
 * current stack frame are of interest:
 *
 * STACK_FRAME  (often called the Frame Pointer or FP) points to the
 *              beginning of the current stack frame,
 *
 * STACK_FRONT  (often called the Stack Pointer or SP) points to the end of
 *              the current stack frame, if another call is made then this
 *              value is copied into STACK_FRAME.
 *
 * these fields may be found in jmp_buf[STACK_FRAME] and jmp_buf[STACK_FRONT]
 *
 * For debugging purposes we define two initial guard values to protect
 * against accidental POPs and RETURNS.
 *
 * INITIAL_FRAME_OFFSET should allow enough space for all values popped by a
 *                      RETURN (it will jump to location 0)
 *
 * INITIAL_FRONT_OFFSET is INITIAL_FRONT_OFFSET plus an allowance for a POP
 *                      to get a zero.
 *
 * These values are expressed as -<no of 32 bit ints> we allow a safety factor
 * of two just to be extra cautious, i.e. if return pops one int we define the
 * INITIAL_FRAME_OFFSET as -2 rather -1.
 */

#ifdef rs6000

#define MULTI_TASKING
#define STACK_SIZE      MIN_STACK_SIZE * 4 /* reserve a very big STACK */
#define STACK_FRAME     0       /* unknown EPK */
#define STACK_FRONT     1       /* EPK */
#define INITIAL_FRAME_OFFSET    (-2)
#define INITIAL_FRONT_OFFSET    (-4)


#include <setjmp.h>
typedef int ansa_RegDump[105] ;  /* we save more register than setjmp */

extern int  stack_dump   (/* ansa_RegDump env */);
```

```
extern void stack_switch (/* ansa_RegDump env, int val */);


#endif  /* rs6000 */

#endif  /* __STACK_INCLUDED__ */
```

- ibm.h
  Zbiór ten należy utworzyć w katalogu master/include/machine. Definicje typów w zbiorze ibm.h są zgodne ze zbiorem sparc.h

```
/* Header Start */
/*
 *  Copyright 1989 APM Limited (ISA Project)
 *
 *  File: ibm.h
 *  Date Created:
 *
 *  General Description:
 *    Machine specific values and constants for hppa.
 *    Need the conditional compilation flags just in case this
 *    header file is compiled within another system (e.g. X11)
 *    which insists on defining its own conflicting types.
 *
 *  $Header: sparc.h,v 1.2 90/08/15 13:26:50 can Exp $
 *
 */
/* Header End */

#ifndef __MACHINE_INCLUDED__
#define __MACHINE_INCLUDED__

/* Various common types */

#ifndef _BASIC_TYPES_
#define _BASIC_TYPES_
typedef int                     ansa_Boolean;
typedef unsigned short int      ansa_ShortCardinal;     /* must be 16 bit */
typedef unsigned long int       ansa_Cardinal;          /* must be 32 bit */
typedef unsigned long int       ansa_LongCardinal;      /* should be 64 bit */
typedef short int               ansa_ShortInteger;      /* 16 bit */
typedef long int                ansa_Integer;           /* 32 bit */
typedef long int                ansa_LongInteger;       /* should be 64 bit */
typedef float                   ansa_Real;
typedef double                  ansa_LongReal;
typedef char                    *ansa_String;
```

```
typedef unsigned char          ansa_Octet;
typedef char                   ansa_Char;
#endif /* _BASIC_TYPES_ */

/*
 *      define LITTLE_ENDIAN if that is the case for this system
 *      VAX, 80x86, ARM
 */
/*#define LITTLE_ENDIAN*/

#endif /* __MACHINE_INCLUDED__ */
```

- rs6000.macros
  Zbiór ten należy utworzyć w katalogu master/util/config . Zbiór zawiera definicję symboli niezbędnych do prawidłowego wygenerowania środowiska do kompilacji.

```
/* Header Start */
/*
 *  Copyright 1989 APM Limited (ISA Project)
 *
 *  File: riscix.macros
 *  Date Created: 03/12/92
 *
 *  General Description:
 *    Common definitions for all versions of
 *
 *  $Header: /disc2/local/src/ANSAware3.0Dist/master/util/config/RCS/riscix.macr
os,v 1.6 90/11/20 17:31:13 njh Exp $
 *
 */
/* Header End */

/**/# platform: $Header: /disc2/local/src/ANSAware3.0Dist/master/util/config/RCS
/riscix.macros,v 1.6 90/11/20 17:31:13 njh Exp $
/**/# operating system: OSName

 BOOTSTRAPCFLAGS = -g -Drs6000
              AS = as
              CC = bsdcc
             CPP = /usr/lpp/X11/Xamples/util/cpp/cpp
            LINT = lint
         INSTALL = /usr/ucb/install -c
              LD = $(CC)
            YACC = yacc
             LEX = lex
             AWK = awk
```

```
             CMP = cmp
            TAGS = ctags
              RM = rm -f
              MV = mv
              CP = cp
           TOUCH = touch
              LN = ln -s
          RANLIB = ranlib
 RANLIBINSTFLAGS = -t
              AR = ar rv
              LS = ls
        LINTOPTS = -axz
     LINTLIBFLAG = -C
            MAKE = make
    STD_IDLFLAGS = $(DEF_IDLFLAGS)
    STD_DPLFLAGS = $(DEF_DPLFLAGS)
 STD_CPP_DEFINES =
     STD_DEFINES = $(DEF_DEFINES) -D__NATIVE_ANSI_INCLUDES__
    STD_INCLUDES = $(DEF_INCLUDES)
STD_IDL_INCLUDES = $(DEF_IDL_INCLUDES)

/* add symbols for extra preprocessor EPK */
CC_CPP_FLAG = -Drs6000 -Dunix

/* define this as you like for normal library compilation */
#define LibraryDefines -D__NATIVE_ANSI_INCLUDES__
#define OptimizedCDebugFlags

#define OperatingSystem         unix
#define OperatingSystemCore     unix
#define UnixCore        YES
#define SysFirstLibraries
#define SysLastLibraries
#define NetworkLibraries

#define OSPlatformFlags -Drs6000
```

- rs6000_AIX_3_1.macros
  Zbiór ten należy utworzyć w katalogu master/util/config. Zbiór zawiera odwołanie do zbioru definiującego symbole niezbędne do prawidłowego wygenerowania środowiska do kompilacji

```
/* Header Start */
/*
 *  Copyright 1989 APM Limited (ISA Project)
 *
 *  File: rs6000_AIX_3_1.macros
```

```
 *  Date Created: 03/12/92
 *
 *  General Description:
 *    Definitions for Acorn RISCiX 1.13.
 *
 *  $Header: /disc2/local/src/ANSAware3.0Dist/master/util/config/RCS/arm_riscix_
1.13.macros,v 1.3 90/11/20 16:47:30 njh Exp $
 *
 */
/* Header End */

/**/# platform version: $Header: /disc2/local/src/ANSAware3.0Dist/master/util/co
nfig/RCS/arm_riscix_1.13.macros,v 1.3 90/11/20 16:47:30 njh Exp $

/*
 * SET VERSION NUMBERS BEFORE MAKING MAKEFILES
 *
 * If you change either of the following numbers, you will need to do a
 * full build (make World) at the top of the sources....
 */
#define AIXPlatform       YES
#define OSName            AIX 3.1
#define OSMajorVersion    3
#define OSMinorVersion    1

#include "rs6000.macros"

#define Architecture    aix
```

2.2 Uzupełnienia w istniejących zbiorach konfiguracyjnych

- Zmiany w zbiorze master/src/ansa/capsule/stack.cpp
  Zbiór ten należy uzupełnić o definicję procedur **stack_dump** i **stack_switch** służących do przełączaniaa stosu. Zostały one dopisane do zbioru stack.cpp z powodu niemożności wykorzystania systemowych procedur **setjmp** i **longjmp**. Procedura **stack_dump** zapamiętuje, a **stack_switch** odtwarza następujące rejestry procesora:

  - rejestry ogólnego przenaczenia GPR
  - rejestry specjalne SPR
  - rejestry zmiennoprzecinkowe FPR

Do zbioru stack.cpp dodano:

```
#if defined(rs6000)

 #----------------------------------------------------------------------------
 # Stack switching routines for ANSAware
 #
 # Date  : 92/02/26
 #
 # Remarks : Maybe we are missing some registers (special purpose / condition)
 #
 #
 # --- name the source-file (for debugger etc.) ----
        .file "my_stack.s"

 # --- declare function entry points ---------------
        .globl .stack_dump[PR]
        .globl .stack_switch[PR]

 # --- declare position of registers in save area
        .set GPR_area,  0
        .set SPR_area,  32 * 4          #--- 32 * 4 = length of GPR_area
        .set FPR_area,  SPR_area + 5*4  #--- 5*4 = length of SPR_area
        .set FPSCR_area, FPR_area + 32 * 8 #- 32 * 8 = length of FPR_area

 # --- first function -------------------------------------------------------
        .csect  .stack_dump[PR]
        stm     0, GPR_area (3)         # store all reg. at env (first argument/
GPR3)

        stfd     0, FPR_area + 0(3)     # save all FPR (not implemented yet)
        stfd     1, FPR_area + 8(3)
        stfd     2, FPR_area + 10(3)
        stfd     3, FPR_area + 18(3)
        stfd     4, FPR_area + 20(3)
        stfd     5, FPR_area + 28(3)
        stfd     6, FPR_area + 30(3)
        stfd     7, FPR_area + 38(3)
        stfd     8, FPR_area + 40(3)
        stfd     9, FPR_area + 48(3)
        stfd    10, FPR_area + 50(3)
        stfd    11, FPR_area + 58(3)
        stfd    12, FPR_area + 60(3)
        stfd    13, FPR_area + 68(3)
        stfd    14, FPR_area + 70(3)
        stfd    15, FPR_area + 78(3)
        stfd    16, FPR_area + 80(3)
```

```
        stfd    17, FPR_area + 88(3)
        stfd    18, FPR_area + 90(3)
        stfd    19, FPR_area + 98(3)
        stfd    20, FPR_area + 100(3)
        stfd    21, FPR_area + 108(3)
        stfd    22, FPR_area + 110(3)
        stfd    23, FPR_area + 118(3)
        stfd    24, FPR_area + 120(3)
        stfd    25, FPR_area + 128(3)
        stfd    26, FPR_area + 130(3)
        stfd    27, FPR_area + 138(3)
        stfd    28, FPR_area + 140(3)
        stfd    29, FPR_area + 148(3)
        stfd    30, FPR_area + 150(3)
        stfd    31, FPR_area + 158(3)
        cror    0xf, 0xf, 0xf          # nop instruction

        mffs    31                     # move contens of FPSCR to FP31
        stfd    31, FPSCR_area(3)      # store FPSCR
        lfd     31, FPR_area + 158(3)  # restore old contens of FP31


        mfmq    31              # move status registers to last four GPR
        mfxer   30
        mflr    29
        mfctr   28
        mfcr    27
        cror    0xf, 0xf, 0xf   # nop instruction

        stm     27, SPR_area(3) # store the last 4 status registers
        cror    0xf, 0xf, 0xf   # nop instruction

        lm      28, 28*4(3)     # set back the last four registers
        cror    0xf, 0xf, 0xf   # nop instruction

        lil     3,0             # load GPR 3 with 0 (return value)

        brl                     # return to address held in link register
        cror    0xf, 0xf, 0xf   # nop instruction

# --- second function ----------------------------------------------
        .csect  .stack_switch[PR]
        lm      27,SPR_area(3)  # load status registers from env
        cror    0xf, 0xf, 0xf   # nop instruction

        mtmq    31
        mtxer   30
```

```
mtlr    29
 mtctr   28
mtcr    27

cror    0xf, 0xf, 0xf           # nop instruction

lfd     31, FPSCR_area (3)      # load FPSCR into FP31
mtfs    31                      # load FPSCR

                        # restor FPR
lfd      0, FPR_area + 0(3)     # save all FPR (not implemented yet)
lfd      1, FPR_area + 8(3)
lfd      2, FPR_area + 10(3)
lfd      3, FPR_area + 18(3)
lfd      4, FPR_area + 20(3)
lfd      5, FPR_area + 28(3)
lfd      6, FPR_area + 30(3)
lfd      7, FPR_area + 38(3)
lfd      8, FPR_area + 40(3)
lfd      9, FPR_area + 48(3)
lfd     10, FPR_area + 50(3)
lfd     11, FPR_area + 58(3)
lfd     12, FPR_area + 60(3)
lfd     13, FPR_area + 68(3)
lfd     14, FPR_area + 70(3)
lfd     15, FPR_area + 78(3)
lfd     16, FPR_area + 80(3)
lfd     17, FPR_area + 88(3)
lfd     18, FPR_area + 90(3)
lfd     19, FPR_area + 98(3)
lfd     20, FPR_area + 100(3)
lfd     21, FPR_area + 108(3)
lfd     22, FPR_area + 110(3)
lfd     23, FPR_area + 118(3)
lfd     24, FPR_area + 120(3)
lfd     25, FPR_area + 128(3)
lfd     26, FPR_area + 130(3)
lfd     27, FPR_area + 138(3)
lfd     28, FPR_area + 140(3)
lfd     29, FPR_area + 148(3)
lfd     30, FPR_area + 150(3)
lfd     31, FPR_area + 158(3)

lm      0,GPR_area(3)   # restor GPRs
                        # maybe we should set GPR3 to val/val + 1 ???

brl                     # jump back
```

```
        cror    0xf, 0xf, 0xf   # nop instruction

#endif
```

- Zmiany w zbiorze ansaware/ACTION
  Do zbioru Action dołączono wiersz opisujący nową platformę sprzętową. Poniżej opisano poszczególne pola tego wiersza:

Pole 1:	rs6000_3_1_5	
Pole 2:	unix	
Pole 3:	srl	
Pole 4:	config.h	(capsule/config.h)
Pole 5:	bsd.h	(capsule/opsys.h)
Pole 6:	ibm.h	(capsule/machine.h)
Pole 7:	aix.h	(capsule/stack.h)
Pole 8:	sparcIEEE.h	(stub/moveMacs.h)
Pole 9:	unix	
Pole 10:	unix	
Pole 11:	IBM RS6000 runing AIX 3.1.5	

Zbiór konfiguracyjny ACTION po modyfikacji:

```
# Header Start
#
#  Copyright 1990 APM Limited (ISA Project)
#
#  File: ACTION
#  Date Created: 15/12/92
#
#  General Description:
#    Configuration file for ANSAware.sh script.
#
#  $Header: /disc2/usr/local/src/ANSAware4.0/ansaware/RCS/ACTION,
v 1.16 1992/03/12 17:58:03 can Exp $
#
# Header End
#
# A # in column 1 indicates a comment
# Each line consists of a set of colon separated fields representing
# the configuration for a given platform.
# The fields are as follows:
```

```
# Field 1: the platform and build directory name
# Field 2: the base name for the build script to be run in the build
#    directory, e.g. unix for unix.sh
# Field 2: the base name for the build script to be run in the build
#    directory, e.g. unix for unix.sh
# Field 3: the base name for the script to create the build shadow tree
# Field 4: the header file to be copied from config to capsule/config.h
# Field 5: the header file to be copied from opsys to capsule/opsys.h
# Field 6: the header file to be copied from machine to capsule/machine.h
# Field 7: the header file to be copied from capsule/stack to capsule/stack.h
# Field 8: the header file to be copied from stub/MoveMacs to stub/moveMacs.h
# Field 9: CORE_* directory to merge into include/capsule
# Field 10: comma separated list of files containing directories to be shadowed
# Field 11: a comment to be shown to the user
# All scripts live in the directory toplevel
#
rs6000_3_1_5:unix:srl:config.h:bsd.h:ibm.h:aix.h:sparcIEEE.h:unix:unix:
 IBM RS6000 runing AIX 3.1.5
sun3_sos_4.0:unix:srl:config.h:bsd.h:680x0.h:sun3_sos.h:680x0IEEE.h:unix:unix:
 SunOS4.0 on a Sun3
sun3_sos_4.1:unix:srl:config.h:bsd.h:680x0.h:sun3_sos.h:680x0IEEE.h:unix:unix:
 SunOS4.1 on a Sun3
sun4_sos_4.0:unix:srl:config.h:bsd.h:sparc.h:sun4_sos.h:sparcIEEE.h:unix:unix:
 SunOS4.0 on a Sun4 Sparc
sun4_sos_4.1:unix:srl:config.h:bsd.h:sparc.h:sun4_sos.h:sparcIEEE.h:unix:unix:
 SunOS4.1.1 on a Sun4 Sparc
sun4_sos_4.0:unix:srl:config.h:bsd.h:sparc.h:sun4_sos.h:sparcIEEE.h:unix:unix:
 SunOS4.0 on a Sun4 Sparc
sun4_sos_4.1:unix:srl:config.h:bsd.h:sparc.h:sun4_sos.h:sparcIEEE.h:unix:unix:
 SunOS4.1.1 on a Sun4 Sparc
drs6000_V.4:unix:srl:config.h:sys5.h:sparc.h:drs6000.h:sparcIEEE.h:unix:unix:
 DRS/NX 6000 on an ICL DRS 6000
arm_riscix_1.13:unix:srl:config.h:bsd.h:arm.h:riscix.h:ARMIEEE.h:unix:unix:
 AcornR140 running RISCiX 1.13
hp300_hpux_7.0:unix:srl:config.h:hpux.h:680x0.h:hp300_hpux.h:680x0IEEE.h:unix:
unix: Bobcats running HPUX7.0
hp300_hpux_8.0:unix:srl:config.h:hpux.h:680x0.h:hp300_hpux.h:680x0IEEE.h:unix:
unix: Bobcats running HPUX8.0
hp400_hpux_8.0:unix:srl:config.h:hpux.h:680x0.h:hp300_hpux.h:680x0IEEE.h:unix:
unix: HP/Apollo 400's running HPUX8.0
hp800_hpux_7.0:unix:srl:config.h:hpux.h:hppa.h:hp800_hpux.h:HPPAIEEE.h:unix:
unix: Spectra running HPUX7.0
hp700_hpux_8.0:unix:srl:config.h:hpux.h:hppa.h:hp800_hpux.h:HPPAIEEE.h:unix:
unix: Snakes running HPUX8.0
vax_ultrix_3.0:unix:srl:config.h:bsd.h:vax.h:vax_ultrix.h:VAX.h:unix:unix:
 Ultrix3.0 on a Vax
mips_ultrix_3.1:unix:srl:config.h:bsd.h:elmips.h:mips_ultrix.h:ELMIPSIEEE.h:
```

```
unix:unix: DECStation 3100 (MIPS) running Ultrix3.1 @
vax_ultrix_3.0:unix:srl:config.h:bsd.h:vax.h:vax_ultrix.h:VAX.h:unix:unix:
 Ultrix3.0 on a Vax
mips_ultrix_3.1:unix:srl:config.h:bsd.h:elmips.h:mips_ultrix.h:ELMIPSIEEE.h:
unix:unix: DECStation 3100 (MIPS) running Ultrix3.1
msdos_dist:msdos_dist:srl_msdos:config.h:msdos.h:80x86.h:msdos.h:80x86IEEE.h:
msdos:msdos_dist: msdos ANSAware distribution
vms_dist:vms_dist:srl_vms:config.h:vms.h:vax.h:vms.h:vax.h:vms:vms_dist:
 vms ANSAware distribution
gen_dist:gen_dist:scp::::::gen_dist: generic ANSAware distribution
```

- Zmiany w zbiorze master/include/capsule/stack/Imakefile
  Dodano wynikającą z nowej platformy sprzętowej zależność od zbioru **aix.h** w linii 27:

```
        aix.h \
```

- Zmiany w zbiorze master/util/config/Imake.rules
  Zmiany w tym zbiorze dotyczą reguł tworzenia bibliotek, i polegają na dostosowaniu do platformy programowej systemu SYSV. W lini 393 definiowane jest makro UpdateLibrary:

```
#define UpdateLibrary(libname,objlist,tfile)                              @@\
libname: objlist tfile                                                    @@\
        $(AR) $@ objlist                                                  @@\
        $(RANLIB) $@
```

  a w linii 400 makro UpdateLibraryUpOne:

```
#define UpdateLibraryUpOne(libname,objlist,tfile)                         @@\
libname: objlist                                                          @@\
        $(AR) $@ objlist                                                  @@\
        $(RANLIB) $@                                                      @@\
        $(TOUCH) tfile
```

- Zmiany w zbiorze master/util/config/Imake.tmpl
  Dodano w lini 73 następujący fragment definiujący środowisko dla systemu RS6000:

```
#if (defined_3_1_5) || (defined rs6000)
#define MacroIncludeFile "rs6000_AIX_3_1.macros"
#define MacroFile rs6000_AIX_3_1.macros
#undef rs6000_3_1_5
#endif /* RS6000 AIX */
```

- Zmiany w zbiorze master/util/imake/imake.c
  W zbiorze dodano definicje zmiennych środowiskowych w linii 106:

```
#if defined(drs6000_V_4) || defined(drs6000)
#define REDUCED_TO_ASCII_SPACE
#define SYSV
#endif
```

- Zmiany w zbiorze master/include/opsys/bsd.h
  Dodano fragment definiujący zmienną środowiskową w linii 194:

```
#if defined(rs6000)
#define YYPARSE_DEFINED_AS_INT
#endif
```

- Zmiany w zbiorze ANSAware.sh
  Ze względu na wymagania wymagania systemu zmieniono linię 207 na:

```
BUILDTB=./BuildAW.sh
```

2.3 Poprawki w zbiorach spowodowane zmianą środowiska

Poniższa grupa poprawek jest zawiązana z dostosowaniem sie do nowego środowiska programowo-sprzętowego.

- Zmiany w zbiorze master/RecoverMakefile.sh:
  polegają na ustawieniu zmiennej środowiskowej BOOTSTRAPCFLAGS w linii 18

```
if make -f Makefile.boot BOOTSTRAPCFLAGS="'-Drs6000_3_1_5 $BOOTSTRAPCFLAGS'" Mas
terMakefile; then
```

- Poprawki w zbiorze master/include/capsule/CORE_unix/mpsipc.h
  Ze względu na występujący konflikt nazw w strukturze **fifo_header** trzeba zmienić nazwę jednego pola, z **fh_len** na **fh_length** w linii 29:

```
    unsigned short fh_length;   /* length of fh_pid + message */
```

- Poprawka w zbiorze master/src/ansa/capsule/CORE_unix/mpsipc.c
  Powyższa poprawka w definicji struktury wymusza automatyczną zmianę nazwy pola struktury w linii 297:

```
    fh->fh_length = cc;
```

i w linii 365:

```
size = fh.fh_length;
```

- Poprawki w zbiorze master/include/capsuled/CORE_unix/mpsipc.h
  Ze względu na występujący konflikt nazw w strukturze **fifo_header** trzeba zmienić nazwę jednego pola, z **fh_len** na **fh_length** w linii 29:

```
unsigned short fh_length;   /* length of fh_pid + message */
```

- Poprawka w zbiorze master/src/ansa/capsuled/CORE_unix/mpsipc.c
  Powyższa poprawka w definicji struktury wymusza automatyczną zmianę nazwy pola struktury w linii 297:

```
fh->fh_length = cc;
```

i w linii 365:

```
size = fh.fh_length;
```

- Poprawki w zbiorze master/include/capsulep/CORE_unix/mpsipc.h
  Ze względu na występujący konflikt nazw w strukturze **fifo_header** trzeba zmienić nazwę jednego pola, z **fh_len** na **fh_length** w linii 29:

```
unsigned short fh_length;   /* length of fh_pid + message */
```

- Poprawka w zbiorze master/src/ansa/capsulep/CORE_unix/mpsipc.c
  Powyższa poprawka w definicji struktury wymusza automatyczną zmianę nazwy pola struktury w linii 297:

```
fh->fh_length = cc;
```

i w linii 365:

```
size = fh.fh_length;
```

- Poprawki w zbiorze master/src/ansa/notify/GrReaper.dpl
  W zbiorze dodano w linii 42:

```
#ifdef rs6000
#include <sys/types.h>
#include <sys/dir.h>
#endif
```

a w linii 137:

```
#ifdef rs6000
        struct direct *dp;
#endif
```

- Poprawka w zbiorze master/include/ansi/AWsocket.h
  Polega na dodaniu w linii 24 zależności od (rs6000):

```
#if defined(_ansa_riscix_) || defined(_ansa_ultrix_) || defined(_ansa_hpux_8_0_)
 || (rs6000)
```

- Poprawka w zbiorze master/include/ansi/AWtime.h
  Polega na dodaniu w linii 22 zależności od (rs6000):

```
#if defined(_ansa_sunos_) || defined(_ansa_drs6000_) || defined(rs6000)
```

Rozdział 3

3.1 Generacja systemu ANSA

Przedstawione uprzednio nowe zbiory konfiguracyjne i poprawki wniesione do zbiorów źródłowych umożliwiają generacje oprogramowania ANSA na komputerach serii RS6000. Poniżej przedstawiono parametry podawane przy instalacji systemu ANSA.

```
ANSAware can be built over the following UNIX platforms:

rs6000_3_1_5: IBM RS60000 runing AIX 3.1.5
sun3_sos_4.0: SunOS4.0 on a Sun3
sun3_sos_4.1: SunOS4.1 on a Sun3
sun4_sos_4.0: SunOS4.0 on a Sun4 Sparc
sun4_sos_4.1: SunOS4.1.1 on a Sun4 Sparc
drs6000_V.4: DRS/NX 6000 on an ICL DRS 6000
arm_riscix_1.13: AcornR140 running RISCiX 1.13
hp300_hpux_7.0: Bobcats running HPUX7.0
hp300_hpux_8.0: Bobcats running HPUX8.0
hp400_hpux_8.0: HP/Apollo 400's running HPUX8.0
hp800_hpux_7.0: Spectra running HPUX7.0
hp700_hpux_8.0: Snakes running HPUX8.0
vax_ultrix_3.0: Ultrix3.0 on a Vax
mips_ultrix_3.1: DECStation 3100 (MIPS) running Ultrix3.1
msdos_dist: msdos ANSAware distribution
vms_dist: vms ANSAware distribution
gen_dist: generic ANSAware distribution

Please enter the desired platform, any of the following are valid:
rs6000_3_1_5
sun3_sos_4.0
sun3_sos_4.1
sun4_sos_4.0
sun4_sos_4.1
drs6000_V.4
arm_riscix_1.13
hp300_hpux_7.0
```

```
hp300_hpux_8.0
hp400_hpux_8.0
hp800_hpux_7.0
hp700_hpux_8.0
vax_ultrix_3.0
mips_ultrix_3.1
msdos_dist
vms_dist
gen_dist


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process
rs6000_3_1_5



Do want to use an alternate shadow tree (i.e. one other than rs6000_3_1_5)?

type <RETURN>           if the default (rs6000_3_1_5) is acceptable
type the name of an alternative shadow tree


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process



Platform has generic type unix
Using build script ./ansaware/unix.bld
Makefile recover script is ./ansaware/unix.rcvr
Using installation script ./ansaware/unix.inst
Re-installation script is ./ansaware/unix.rinst
Creating shadow tree ./rs6000_3_1_5 for ./rs6000_3_1_5, this may take a few minu
tes

Shadow tree ./rs6000_3_1_5 built
./ansaware/unix.bld -> rs6000_3_1_5/./BuildAW.sh
./ansaware/unix.rcvr -> rs6000_3_1_5/RecoverMakefile.sh
./ansaware/unix.inst -> rs6000_3_1_5/InstallAWBuild.sh
./ansaware/unix.rinst -> rs6000_3_1_5/ReInstallAWBuild.sh
Linking header files ...
include/capsule/config.h == include/config/config.h
include/capsule/opsys.h == include/opsys/bsd.h
include/capsule/machine.h == include/machine/ibm.h
include/capsule/stack.h == include/stack/aix.h
include/stub/MoveMacs.h == include/stub/MoveMacs/ARMIEEE.h
```

```
include/stub/m_builtin.h == include/stub/MoveMacs/m_ARMIEEE.h
... Done header files

Merging include/capsule/CORE_unix into include/capsule ...
include/capsule/CORE_unix/gpin.h == include/capsule/gpin.h
include/capsule/CORE_unix/mpsipc.h == include/capsule/mpsipc.h
include/capsule/CORE_unix/schedule.h == include/capsule/schedule.h
include/capsule/CORE_unix/sysdep.h == include/capsule/sysdep.h
include/capsule/CORE_unix/task.h == include/capsule/task.h
include/capsule/CORE_unix/gpin.hd == include/capsule/gpin.hd
include/capsule/CORE_unix/mpsipc.hd == include/capsule/mpsipc.hd
include/capsule/CORE_unix/mpsmsnl.hd == include/capsule/mpsmsnl.hd
include/capsule/CORE_unix/mpstcp.hd == include/capsule/mpstcp.hd
include/capsule/CORE_unix/mpsudp.hd == include/capsule/mpsudp.hd
include/capsule/CORE_unix/schedule.hd == include/capsule/schedule.hd
include/capsule/CORE_unix/sysdep.hd == include/capsule/sysdep.hd
include/capsule/CORE_unix/system.hd == include/capsule/system.hd
include/capsule/CORE_unix/task.hd == include/capsule/task.hd

Merging include/ansi into include/capsule ...
include/ansi/AWdirent.h == include/capsule/AWdirent.h
include/ansi/AWfcntl.h == include/capsule/AWfcntl.h
include/ansi/AWinet.h == include/capsule/AWinet.h
include/ansi/AWmacros.h == include/capsule/AWmacros.h
include/ansi/AWmath.h == include/capsule/AWmath.h
include/ansi/AWsocket.h == include/capsule/AWsocket.h
include/ansi/AWstdarg.h == include/capsule/AWstdarg.h
include/ansi/AWstdio.h == include/capsule/AWstdio.h
include/ansi/AWstdlib.h == include/capsule/AWstdlib.h
include/ansi/AWstring.h == include/capsule/AWstring.h
include/ansi/AWtime.h == include/capsule/AWtime.h
include/ansi/AWunistd.h == include/capsule/AWunistd.h

Merging include/lib into include/capsule ...
include/ansi/Xtpin.h == include/capsule/Xtpin.h
include/ansi/chkptlib.h == include/capsule/chkptlib.h
include/ansi/kbdos.h == include/capsule/kbdos.h
include/ansi/kbinput.h == include/capsule/kbinput.h
include/ansi/kbvms.h == include/capsule/kbvms.h
include/ansi/reloclib.h == include/capsule/reloclib.h
include/ansi/reloclib.hd == include/capsule/reloclib.hd
... Done directory merge

Building for platform rs6000_3_1_5 in rs6000_3_1_5

This is the user configuration section of the ANSAware installation.
You are building for rs6000_3_1_5 in the directory rs6000_3_1_5
```

```
You must have read the installation and configuration section of the
ANSAware Implementation Manual before proceeding.
If you have not done so, or if you do not understand any
of the questions which follow please type *ABORT* in response to any question.

Please enter the full pathname of the root directory in which the ANSAware
source tree is installed, i.e. the directory this script is in and the
one you read the distribution tape in.
If this directory is to be NFS served then this pathname should be
accessible from all NFS clients.
Similarly pathnames specified for NFS served ANSAware installation points
must be accessible from at least all of the NFS clients which are of the
same machine type.

THERE IS NO DEFAULT!!!



type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process
/home/genda/ar6





ANSAware requires that two installation points be specified,
the first of which is used to install read only files such as program,
header and library files, whilst the second
is used for the installation of the various read/write logging files used by
ANSAware. These two directories may of course be identical.

The following subdirectories are used below the
read only installation points:
bin
include
lib
etc
and below the read/write installation point we have:
etc

In addition a suffix may be specified for each of these
sets of subdirectories.
If the suffix for the read only directories were to be ansa the installation
would place files in the subdirectories:
bin/ansa
include/ansa
```

```
lib/ansa
etc/ansa

Please enter the full pathname of the root directory in which the ANSAware
read only files are to be installed,
the default is /home/genda/ar6/install/rs6000_3_1_5

type <RETURN>           if the default is acceptable


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process



Please specify the suffix to be appended to these read-only subdirectories,
the default is to use a NULL, i.e. no suffix.

type <RETURN>           if the default is acceptable


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process



Please enter the full pathname of the root directory in which the ANSAware
read/write files are to be installed,
the default is /home/genda/ar6/install/rs6000_3_1_5

type <RETURN>           if the default is acceptable


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process



Please specify the suffix to be appended to these read/write subdirectories,
the default is to use a NULL, i.e. no suffix.

type <RETURN>           if the default is acceptable


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process
```

```
The ANSAware read only installation root directory is /home/genda/ar6/install/rs
6000_3_1_5
The ANSAware read only installation subdirectories are:
/home/genda/ar6/install/rs6000_3_1_5/bin
/home/genda/ar6/install/rs6000_3_1_5/etc
/home/genda/ar6/install/rs6000_3_1_5/lib
/home/genda/ar6/install/rs6000_3_1_5/include

The ANSAware read/write installation root directory is /home/genda/ar6/install/r
s6000_3_1_5
The ANSAware read/write installation subdirectories are:
/home/genda/ar6/install/rs6000_3_1_5/etc

Please enter the MPS modules which you wish to be configured into your
system.  Any set of the following are valid:
        IPC UDP TCP

The default is IPC UDP

type <RETURN>           if the default is acceptable


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process
UDP TCP



The MPS modules selected are: UDP TCP


Please enter the hostname of the master trader host.  Case IS significant.
The default master trader host is rsd

type <RETURN>           if the default is acceptable


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process



The master trader host is rsd

Please enter the IP address for rsd.  This is used to construct
```

```
the domain identifier for the master trading domain.

The address is of the form a.b.c.d
where a, b, c, d are integers between 0 and 255, inclusive.

THERE IS NO DEFAULT!!!

type *ABORT*<RETURN>    to abort the installation process
type a.b.c.d<RETURN>    for the master trading host's IP address


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process
149.156.99.53



The master trader's IP address is 149.156.99.53

Please enter the hostname of the local trader host.  Case IS significant.
The default local trader host is rsd

type <RETURN>           if the default is acceptable


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process


The local trader host is rsd


The local trader's IP address is 149.156.99.53

ANSAware uses some commonly available features from ANSI C e.g.
void, #if defined(foo), #elif.  Where possible it can also use other features
such as function prototypes.  The next two questions enable you to specify an
alternate C compiler and/or C pre-processor:

Enter the pathname of an alternate C compiler (e.g. /usr/bin/gcc)

type <RETURN>           if the default (i.e. /bin/cc) is acceptable


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process
/bin/bsdcc
```

```
The C compiler is /bin/bsdcc

Enter the /bin/bsdcc command line specification for any other compiler options
required by /bin/bsdcc, for example to select an alternate pre-processor.

type <RETURN>           if the defaults used by the compiler is acceptable


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process
-Drs6000 -Dunix




Do want to build profiled libraries for use with gprof.
They cannot be used on some systems and their use will be
suppressed even if you ask for them here.
If so type Y or y, otherwise they will not be built.


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process


Please confirm the following build parameters are OK

ANSAware to be built for platform       rs6000_3_1_5
ANSAware to be built in shadow tree     rs6000_3_1_5
ANSAware read-only installation root directory is /home/genda/ar6/install/rs6000
_3_1_5
ANSAware read/write installation root directory is /home/genda/ar6/install/rs600
0_3_1_5
Master trader host is                   rsd     (IP address 149.156.99.53)
Trader host is                          rsd     (IP address 149.156.99.53)
Use the following MPS modules           UDP TCP
The C compiler used will be             /bin/bsdcc
The options selected for /bin/bsdcc are -Drs6000 -Dunix
The profiled libraries will be NOT built

If this is not OK type N or n, otherwise the build will continue


type your reply, followed by <RETURN>
type *ABORT*<RETURN>   to abort the installation process
```

W tak wygenerowanym środowisku oprogramowania ANSA wykonano serię testów zaproponowannych w rozdziale 7 dokumentacji ANSAware 4.0 System Manager's Guide. Przeprowadzone testy wykazały poprawne działanie wygenerowanego oprogramowania.