074 A

PRZEMYSŁOWY INSTYTUT AUTOMATYKI I POMIARÓW
MERA-PIAP
Al. Jerozolimskie 202 02-222 Warszawa Telefon 23-70-81

Ośrodek Automatyzacji Procesów Produkcji (OAP)

Pracownia Oprogramowania Cyfrowych Systemów Automatyki

Główny wykonawca mgr inż. Stanisław Wółtański

Wykonawcy mgr inż. Kazimierz Maliszewski, mgr inż. Stanisław Wółtański

Konsultant mgr inż. Wojciech Nikiel

Nr zlecenia RP-16.1

Wersja systemu SIRTOS do sterownika grupy robotów i ESP.
Etap 1. Opracowanie i uruchomienie monitora operatorskiego czasu rzeczywistego systemu R-T SIRTOS. Dołączenie monitora wraz z deaseblerem do systemu w małym modelu pamięci.

Zleceniodawca CPBR 7.1.

Pracę rozpoczęto dnia lipiec 1990 **zakończono dnia** 90.09.15

Kierownik Pracowni
Dąbrowska
mgr inż. B.Dąbrowska

Z-ca Dyrektora d/s Automatyki i Pomiarów
doc.dr inż. T.Gałązka

Kierownik Ośrodka
dr inż. M.Wrzesień

Praca zawiera:

stron 19
rysunków
fotografii
tabel 1
tablic
załączników

Nr rejestr. 6497

Rozdzielnik - ilość egz: 3

Egz. 1 BOINTE
Egz. 2 OAP-5
Egz. 3 OAE-4
Egz. 4
Egz. 5
Egz. 6

Analiza deskryptorowa

SYSTEM OPERACYJNY ~~CZASU RZECZYWISTEGO~~
MONITOR ~~OPERATORSKI~~ EKRANOWY
STEROWANIE PROCESEM PRZEMYSŁOWYM

Analiza dokumentacyjna

Praca zawiera opis dyrektyw monitora operatorskiego SRTS systemu operacyjnego SIRTOS w wersji dla "małego modelu pamięci" wraz z przykładami ich użycia.

Tytuły poprzednich sprawozdań

681.5 Technika sterowania automatycznego

UKD

MERA-PIAP/TW 331/78 5000

SPIS TREŚCI Strona

1. Wstep. Przeznaczenie monitora operatorskiego R-T

W pracach nad uruchamianiem i testowaniem zarówno sprzętu jak i oprogramowania jest niezbędny minimalny program, rezydujący zazwyczaj w pamięci EPROM, noszący często miano monitora operatorskiego. Program taki zawiera podstawowe operacje pozwalające odczytywać lub modyfikować pamięć sterownika, uruchamiać testy, odczytywać i zapisywać informację z/do portów we/wy, itp. W przypadku uruchamiania oprogramowania pracującego pod wielozadaniowym systemem czasu rzeczywistego niezbędne są jednak pewne dodatkowe mechanizmy ułatwiające użytkownikowi testowanie całej aplikacji. Takie mechanizmy posiada monitor SRTS. Program SRTS służy do uruchamiania zadań użytkowych i procedur obsługi przerwań pracujących pod wielozadaniowym systemem operacyjnym R-T SIRTOS. Z punktu widzenia systemu operacyjnego monitor jest traktowany jako obsługa przerwania (w zależności od sytuacji - programowego INT n albo sprzętowego, od klawiatury terminala: CTRL-C). Jeżeli monitor nie jest aktywny, to nie ingeruje w pracę systemu (w szczególności nie powoduje żadnych zmian w istniejących zależnościach czasowych). Wywołanie monitora powoduje zablokowanie układu przerwań i wstrzymanie pracy wszystkich zadań do momentu użycia dyrektyw M, N lub E. Nie można oczywiście wstrzymać procesów poza komputerem (np. przychodzących przerwań zewnętrznych); powoduje to nieuniknione zmiany w istniejących zależnościach czasowych. Wady tej nie można niestety w żaden sposób (i w żadnym z podobnych programów) uniknąć. Dzięki wykorzystywaniu przez program monitora mechanizmów systemu operacyjnego istnieje natomiast łatwa możliwość śledzenia pracy systemu, której nie mogą zapewnić monitory uniwersalne np. śledzenie bloku kontrolnego zadania, stosów zadań, startowanie zadań dyrektywą monitora, itp. Monitor SRTS zgłasza się po inicjacji zasobów komputera i systemu operacyjnego a przed uruchomieniem zadań użytkowych oraz na żądanie operatora - po wciśnięciu klawiszy CTRL-C. Użytkownik-programista ma możliwość umieszczania w swoich zadaniach wywołania monitora jako przerwania programowego o numerze zależnym od wygenerowanej aplikacji (możliwość tę należy jednak wykorzystywać w uzasadnionych przypadkach).

2. Komendy monitora operatorskiego SRTS

W opisie komend przyjęto następujące oznaczenia:

segm:offs	- adres heksadecymalny w postaci segment:offset,
count	- liczba heksadecymalna z przedziału 0000÷FFFF,
addr	- liczba heksadecymalna z przedziału 0000÷FFFF traktowana jako adres bezwzględny,
byte	- liczba heksadecymalna z przedziału 00÷FF,
arg	- liczba heksadecymalna jedno lub dwuznakowa traktowana jako bajt albo trzy lub czteroznakowa traktowana jako słowo,
task	- nazwa zadania (identyfikator - aktualnie przyjęto trzyznakową nazwę-symbol, składającą się z dużych liter),
[...]	- element opcjonalny, gdy zostanie pominięty, to przyjmuje wartość domyślną, zależną od typu dyrektywy.

Komendy monitora są jedno- lub dwuliterowe. Po komendzie może wystąpić ciąg parametrów oddzielony przecinkami. Akceptowane są zarówno duże jak i małe litery. Monitor nie wymaga wprowadzania nie znaczących zer, nie akceptuje jednak spacji występujących w parametrach. Przy wykonywaniu komendy monitora podany w parametrach segment nie jest zmieniany, natomiast offset jest obliczany modulo 10000 (heksadecymalnie) - 64 kB. Wszystkie parametry liczbowe wyprowadzane przy dezasemblacji pamięci mają postać heksadecymalną, niezależnie od tego czy są zakończone literą "h", czy nie. Obecnie jest dostępny podany niżej zestaw dyrektyw; w zależności od potrzeb użytkowników istnieje możliwość rozszerzenia ich repertuaru.

B

Ustawienie pułapki programowej (ang. breakpoint). Komenda jest bezparametrowa. Powoduje wyświetlenie na ekranie trzech wierszy:

```
   B0        B1        B2        B3        B4        B5   ...
0000:0000 0000:0000 0000:0000 0000:0000 0000:0000 0000:0000...
_
```

Kursor zostaje ustawiony w pierwszej kolumnie trzeciego wiersza w oczekiwaniu na wprowadzenie adresu pułapki w postaci "[segm:]offs". Jeśli "segm:" nie wystąpi jawnie, to przyjmuje bieżącą wartość rejestru segmentowego CS. Wprowadzenie adresu pułapki powoduje zmianę jej dotychczasowej wartości i przejście kursora na następną pozycję. Możliwe jest jednoczesne ustawienie do ośmiu niezależnych pułapek. Przesuwanie kursora możliwe jest również przez wciśnięcie strzałki w prawo → lub w lewo ←. Powoduje to przeskoczenie przez kursor jednego adresu pułapki zgodnie z kierunkiem strzałki. Po dojściu do skrajnych pozycji kursor przeskakuje na przeciwny koniec wiersza. Wpisanie adresu "0:0" powoduje skasowanie ustawionej pułapki. Wpisanie litery Z kasuje wszystkie ustawione pułapki od pozycji kursora do końca wiersza. Wciśnięcie klawicza CR bez wpisania żadnego znaku kończy ustawianie pułapek. Po dojściu programu do adresu z ustawioną pułapką następuje automatyczne wejście do monitora z podaniem adresu i numeru pułapki.

C[seg1:]off1,[seg2]:off2[,count]

Kopiowanie obszaru pamięci. Adres "seg1:offs1" określa od jakiego adresu kopiujemy, adres "seg2:offs2" na jaki kopiujemy. Jeżeli "seg1:" nie zostanie podany, zastaje nadana mu bieżąca wartość rejestru DS. Brak "seg2:" powoduje nadanie mu wartości "seg1:". Parametr "count" określa liczbę kopiowanych bajtów. Jeżeli nie zostanie on określony - kopiowany jest jeden bajt. Nie jest sprawdzane pokrywanie się kopiowanych obszarów. Jeżeli adres fizyczny "seg1:off1 + count" ≥ "seg2:off2", to wystąpi nie sygnalizowany błąd.

D[segm:]offs[,count]

Wyświetlanie obszaru pamięci w kodzie HEX i kodzie ASCII. Pominięcie "segm:" powoduje nadanie mu bieżącej wartości rejestru segmentowego DS. Parametr "count" określa liczbę wyświetlanych bajtów. Domyślnie przyjmuje on wartość "10" (heksadecymalnie).

E

Wyjście z monitora operatorskiego (powrót do przerwanego programu).

F[segm:]offs,arg[,count]

Wypełnianie obszaru pamięci stałą wartością. Pominięcie parametru "segm" powoduje nadanie mu bieżącej wartości rejestru segmentowego DS. Stała "arg" -w zależności od liczby znaków- traktowana jest jako bajt albo słowo. Licznik "count" określa liczbę powtórzeń argumentu. Domyślnie przyjmuje on wartość "1".

Gtask

Uruchomienie zadania o identyfikatorze "task" (operacja START).

[count]Iaddr

Odczyt bajtu z portu o podanym adresie. Odczytana wartość zostaje wypisana na ekranie w interpretacji kolejno "unsigned dec", "hex" i "ascii". Licznik "count" określa liczbę powtórzeń instrukcji. Domyślnie przyjmuje on wartość "1".

[count]Jaddr

Odczyt słowa z portu o podanym adresie. Odczytana wartość zostaje wypisana na ekranie w interpretacji "unsigned dec", "hex" i "ascii". Licznik "count" określa liczbę powtórzeń instrukcji. Domyślnie przyjmuje on wartość "1".

Ktask

Przerwanie pracy zadania o identyfikatorze "task" (operacja KILL).

L[segm:]offs[,count],arg

Wyszukiwanie bajtu albo słowa. Wybrana sekwencja jest poszukiwana od adresu "segm:offs". Zakres poszukiwania określa licznik bajtów "count". Program podaje adresy wszystkich wystąpień słowa albo bajtu w podanym zakresie. Parametr "arg" definiuje poszukiwaną sekwencję (podaną jako liczba heksadecymalna). W zależności od liczby znaków jest on traktowany jako bajt albo słowo (w razie potrzeby uzupełniany zerami z lewej strony). Jeżeli "segm:" nie jest podany, to przyjmuje bieżącą wartość rejestru segmentowego CS. Pominięcie licznika "count" powoduje nadanie mu wartości "40" (heksadecymalnie).

4

M

MB

MT

Są to komendy o znaczeniu takim jak odpowiednio: N, NB i NT, ale bez wyświetlania zawartości rejestrów przed listowanymi instrukcjami.

N

Przejście monitora w tryb pracy krokowej. Wypisuje się zawartość rejestrów i aktualną instrukcję do wykonania, wskazaną przez parę CS:IP. Jeżeli instrukcja działa na danej w pamięci, wyprowadza się jej zawartość. Dla instrukcji działających na stosie wyprowadza się zawartość szczytu stosu. Dla skoków warunkowych wyprowadza się aktualną wartość warunku w postaci słowa "TRUE" (warunek skoku spełniony) lub "FALSE" (nie spełniony). W pewnych przypadkach wyprowadza się informację diagnostyczną.
Wszystkie wyświetlane dane opisują stan poprzedzający wykonanie instrukcji i mogą być ewntualnie zmodyfikowane przez operatora. Dopiero naciśnięcie klawisza CR powoduje wykonanie aktualnej instrukcji i wyświetlenie następnej. Naciśnięcie innego klawisza powoduje wyjście z pracy krokowej bez wykonania aktualnej instrukcji.
W czasie pracy krokowej ustawione pułapki nie są aktywne.

NB

Przejście do pracy krokowej w trybie śledzenia instrukcji skokowych (ang. branch). Instrukcje skokowe są rozumiane w szerokim sensie jako instrukcje zmieniające sekwencję wykonania programu. Klasa ta obejmuje skoki bezwarunkowe, skoki warunkowe (w tym instrukcje LOOP, LOOPE i LOOPNE), instrukcje CALL, RET, INT, INTO i IRET. Monitor ponadto śledzi i sygnalizuje obecność "podejrzanych" instrukcji (zob. opis komendy NT), gdyby takie wystąpiły.
Na wstępie wyświetlane są rejestry i aktualna instrukcja do wykonania (niekoniecznie skokowa), jak dla komendy N. Naciśnięcie klawisza CR powoduje wykonanie aktualnej i wszystkich dalszych instrukcji aż do napotkania instrukcji skokowej, która jest wyświetlana. Kolejne naciśnięcie klawisza CR powoduje jej wykonanie i przejście bez wydruków

do następnej instrukcji skokowej, itd. Naciśnięcie innego klawisza kończy ten tryb pracy.

NT

Przejście w tryb wykonania programu ze śledzeniem formalnej poprawności instrukcji (test). Na wstępie wyświetla się rejestry i aktualną instrukcję do wykonania - jak dla komendy N. Naciśnięcie klawisza CR powoduje uruchomienie programu pod kontrolą monitora. Program wykonuje się znacznie wolniej niż normalnie, ale bez wyświetlania informacji i zatrzymań dla interwencji operatora, dopóki nie wystąpi "podejrzana" instrukcja. W takim przypadku wyświetla się rejestry i instrukcję i oczekuje na jej akceptację przez operatora. Naciśnięcie CR powoduje kontynuację testu, naciśnięcie innego klawisza - wyjście z tego trybu. Ponowne wywołanie monitora klawiszami CTRL-C również kończy komendę NT i trzeba go użyć dla zakończenia badania programu, który się nie zatrzymuje, ponieważ jest formalnie poprawny.

Określa się następujące rodzaje "podejrzanych" instrukcji:

1. Instrukcje osiągające lub przecinające granice 64 kB-owych aktualnych segmentów programu.
   Offset osiągający lub przekraczający 64 kB jest przez procesor "zawijany", tj. redukowany o 64 kB. Część instrukcji lub danej wykraczająca poza granicę bloku jest odczytywana (lub zapisywana) na początku bloku. Monitor poprzedza taką instrukcję 'komunikatem:
   '****** 64kB limit. Uncertain result:'
   który oznacza, że wynik zarówno dezasemblacji, jak i wykonania instrukcji jest niepewny.
   Instrukcje, po których bezpośrednio występuje koniec aktualnego segmentu kodu są podkreślane komunikatem:
   '------ 64kB CS limit here ------'
   Można go zignorować w przypadku efektywnych skoków bez powrotu.
2. Instrukcje niestandardpwe.
   Pewne instrukcje mają oprócz głównego kodu, który w zasadzie powinien być wygenerowany przez translator, także inne, alternatywne kody. Te dodatkowe kody nazywamy tutaj niestandardowymi. Różnią się one od standardowych w części

zawartej w drugim bajcie instrukcji. Doświadczalna reguła, dotycząca tylko procesorów 8086/88, pozwala traktować zamiennie instrukcje niestandardowe ze standardowymi. Monitor oznacza niestandardowe instrukcje przyrostkiem 'nstd' po 3 znakach nazwy, np.:

```
AND nstd BYTE PTR [1234],27h
```

3. Instrukcje o niedozwolonej postaci.
   Niektóre instrukcje wielobajtowe muszą mieć w określony sposób zgodną zawartość kolejnych bajtów. Np. daleki skok pośredni musi określać, że jego 2-słowowy adres przeznaczenia mieści się w pamięci, a nie w rejestrach. Jeżeli instrukcja jest wewnętrznie niezgodna, to monitor zaopatruje ją komentarzem 'Invalid form'. Wykonanie takiej instrukcji da wynik nieokreślony (przynajmniej tutaj).
4. Instrukcje o nie używanych kodach.
   Kody nie używane zarówno przez procesory 8086/8088, 80186/80188, jak i 80286 są dezasemblowane jako 1-bajtowe instrukcje o nazwie '???' bez operanda. Jeżeli kod w pierwszym bajcie jest prawidłowy, ale jego przedłużenie w drugim bajcie ma wartość nie używaną, to nazwa instrukcji jest '???', a operand jest prawidłowy.
5. Instrukcje mikroprocesora Intel 80286 (iAPX 286).
   Instrukcje specyficzne dla procesora 286 są listowane w postaci pierwszego bajtu i nazwy '286code' bez operanda. Dodaje się komentarz 'Possibly >1 byte' ('Być może >1 bajt') ostrzegający, że monitor nie zna faktycznej długości i postaci instrukcji.Przy pracy na procesorze 8086/8088 należy te instrukcje traktować jako nie używane. Procesor 80286 realizuje ponadto wszystkie instrukcje procesorów 80186/80188, a więc i 8086/8088.
6. Instrukcje mikroprocesorów Intel 80186/188 (iAPX186/188).
   Instrukcje specyficzne dla procesorów 80186/80188 są listowane w postaci pierwszego bajtu, nazwy '186code' i komentarza 'Possibly >1 byte' o znaczeniu podanym wyżej. W pewnych wyjątkowych wypadkach listuje się prawidłową ilość bajtów i po nazwie '186code' następuje operand. Przy pracy na procesorze 8086/8088 należy te instrukcje uważać za nie używane. Procesory 80186/80188 realizują ponadto wszystkie instrukcje procesorów 8086/8088.

Prawidłowo załadowany i nieuszkodzony program utworzony przez standardowy kompilator lub asembler procesora 8086 nie powinien zawierać żadnego z wymienionych rodzajów instrukcji.

[count]Oaddr,arg

Wysłanie bajtu albo słowa "arg" na podany adres portu. Licznik "count" określa liczbę powtórzeń instrukcji. Domyślnie przyjmuje on wartość "1".

Ptask

Wyświetlenie obszaru stosu zadania użytkowego o identyfikatorze "task".

R

Wyświetlenie i zmiana rejestrów. Na ekranie jest wyświetlany wiersz z wartościami rejestrów procesora. Kursor jest ustawiany pod nazwą rejestu, którego wartość ma być zmieniona. Możliwe jest cykliczne przesuwanie kursora w wierszu za pomocą strzałek w prawo → lub w lewo ←. Wciśnięcie "pustego" znaku karetki kończy modyfikację rejestrów i powoduje zgłoszenie się monitora SRTS.

S[segm:]offs

Modyfikacja zawartości pamięci. W jednym wierszu zostaje wyświetlona zawartość 16 bajtów pamięci (w postaci identycznej jak przy dyrektywie D). Kursor zostaje ustawiony w następnym wierszu pod podanym adresem w oczekiwaniu na wprowadzenie nowej zawartości pamięci, podawanej za pomocą kodu "hex" bajtami albo słowami. Wprowadzenie nowej wartości powoduje automatycznie przejście kursora o bajt lub słowo w prawo (rozróżnienie dokonywane jest na podstawie liczby wprowadzonych znaków). W ramach wiersza można zmieniać położenie kursora za pomocą strzałek w prawo → lub w lewo ←. Próba przejścia strzałką w prawo poza wiersz powoduje wyświetlenie nowego wiersza. Wciśnięcie "pustego" znaku karetki powoduje zakończenie modyfikacji rejestrów i zgłoszenie się monitora. Domyślnie "segm:" przyjmuje bieżącą wartość rejestru segmentowego DS.

11

T

Wyświetlenie bloku kontrolnego zadania (TCB). Jego zawartość stanowią następujące parametry: stan zadania, adres wejścia do zadania, początek stosu, bieżący wskaźnik stosu, adres semafora albo bufora cyklicznego, licznik taktów zegara czasu rzeczywistego (100 ms), cykl restartu zadania, licznik czasu w cyklu i wskaźnik wejścia do koordynatora (dokładne znaczenie tych parametrów zostało podane w opisie systemu operacyjnego SIRTOS, por. np.: Biuletyn MERA-PIAP nr 4/144 z 1989 roku). Dodatkowo dla zadań oczekujących na semaforze jest podawana bieżąca oraz oczekiwana wartość semafora.

[count]U[segm:]offs[,B]

Dezasemblacja programu od adresu "segm"offs". Instrukcje są listowane w postaci heksadecymalnej, symbolicznej i znakowej (w zakresie drukowalnych znaków ASCII). Jeżeli parametr "segm:" nie wystąpi jawnie, to przyjmuje wartość bieżącą rejestru segmentowego CS. Licznik "count" ogranicza długość analizowanego pola pamięci (w bajtach). Jeżeli nie jest on podany,to jest dekodowana tylko jedna instrukcja. Jeżeli ostatni bajt obszaru nie kończy jakiejś instrukcji, monitor wyprowadza także dalsze bajty aż do jej końca. Opcja "B" oznacza, że mają być listowane tylko instrukcje powodujące zmianę sekwencji wykonania programu (ang. branch; zob. opis komendy NB).

+,-,*,/

Arytmetyka na adresach heksadecymalnych (dodawanie odejmowanie, mnożenie i dzielenie z resztą). Wszystkie obliczenia wykonywane są na liczbach nieujemnych. Wynik obliczeń jest podawany modulo 10000 (heksadecymalnie).

CR (karetka)

Przewinięcie ekranu o dwa wiersze w górę i zgłoszenie monitora symbolem "SRTS>" - w oczekiwaniu na następną komendę operatora.

3. Generacja wersji monitora operatorskiego

W aktualnej wersji 2.3 systemu SIRTOS jest on konsolidowany, podobnie jak jądro, z zadaniami użytkowymi. Dołączenie monitora SRTS jest uwarunkowane nadaniem wartości "1" symbolowi MSRTS znajdującemu się w zbiorze "exm.h" i dołączeniem do konsolidacji zbiorów: "msrts.o", "mints.o" i "mon.o". Prezentowana wersja monitora operatorskiego jest przeznaczona do stosowania w aplikacjach systemu SIRTOS kompilowanych i konsolidowanych w tzw. "małym modelu pamięci" (64kB kodu i 64 kB danych). Dodatkową opcją monitora jest możliwość dezasemblacji zawartości pamięci zarówno EPROM-u jak i RAM-u. Opcja ta jest aktywna (zob. opis dyrektywy "u"), gdy symbolowi DEA w zbiorze "msr.h" ~~przyporządkujemy~~ przypiszemy wartość "1" oraz dokonsolidujemy zbiór "dasm.o". W przypadku potrzeby użycia monitora w wersji minimalnej przypisujemy symbolowi MEM w zbiorze "exm.h" wartość "0" - nie są wówczas dostępne zazwyczaj rzadziej wykorzystywane komendy: "c", "f", "m" ("mb", "mt"), "l" oraz operacje "+", "-", "*" i "/". Przykładowa postać zbioru "makefile" systemu AZTEC C do kompilacji, konsolidacji i konwersji na kod INTEL-HEX systemu SIRTOS z monitorem SRTS w pełnej konfiguracji wygląda następująco:

```
*=sirtos.
R=mtest.o srom.o
S=srts.o
A=inits.o proc.o msrts.o mints.o mon.o dasm.o
U=uinit.o uints.o usersfs.o
W=user.o
I=vi.o
T=tkbd.o tmon.o tprt.o tmmc.o
D=tbgt.o ram.o tads.o tmw32.o
M=-c 500 -d 3
K=22

$*hex: $*exe
@hex86 -z -s$K $*.exe
@echo SIRTOS OK!
@echo nu makefile end
```

```
$*exe: $I $R $S $A $U $W $T $D
@ln -T $M -o $@ $I $R $S $A $U $W $T $D -lc

$S:
cc +F $*

srts.o:     gcc.h gsc.h gse.h ees.h

msrts.o:    gcc.h gsc.h gse.h ees.h guc.h       gic.h gfe.h exm.h msr
mints.o:    gcc.h                                                msr

vi.o:       gcc.h                                      gfe.h exm.h
uinit.o:    gcc.h gsc.h gse.h ees.h       gue.h gic.h gfe.h exm.h
uints.o:    gcc.h gsc.h gse.h ees.h guc.h gue.h gic.h       exm.h
usersfs.o:                            guc.h       gic.h       exm.h

tkbd.o:     gcc.h             ees.h guc.h gue.h
tmon.o:     gcc.h             ees.h guc.h gue.h gic.h
tprt.o:     gcc.h             ees.h guc.h gue.h gic.h
tmmc.o:     gcc.h             ees.h guc.h gue.h             mmc.h
tbgt.o:     gcc.h             ees.h       gue.h       gfe.h
ram.o:      gcc.h             ees.h       gue.h
tads.o:     gcc.h             ees.h       gue.h
```

Zbiór "exm.h" dla powyższej przykładowej aplikacji ma postać:

```
/*~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                Warunkowe sterowanie kompilacja: "exm.h"
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~*/
#define MSRTS 1 /* Czy jest monitor operatorski SRTS? 1-tak; 0-nie */
#define DEA   1 /* z dezasemblacja (DEA=1) lub bez (DEA=0)          */
#define MEM   1 /* z (MEM=1) lub bez (MEM=0) "c", "l", "f", "m"     */

extern void msrts () ;    /* wywolanie monitora z przerwania progr. */
extern void srts () ;     /* glowna funkcja monitora operatorskiego */
```

4. Przykłady wykorzystania dyrektyw monitora SRTS

Podane dalej przykłady ilustrują działanie najczęściej używanych dyrektyw monitora operatorskiego.

```
    B0          B1          B2          B3          B4          B5          B6          B7
0500:0345 0500:0678 0500:1234 0000:0000 0000:0000 0000:0000 0000:0000 0000:0000

SRTS>b

    B0          B1          B2          B3          B4          B5          B6          B7
0500:0345 0500:0678 0500:1234 0000:0000 0000.0000 0000:0000 0000:0000 0000:0000

SRTS>b

    B0          B1          B2          B3          B4          B5          B6          B7
0500:0345 0500:0678 0500:5678 0500:0AAA 0500:0BBB 0000:0000 0000:0000 0000:0000

SRTS>b

    B0          B1          B2          B3          B4          B5          B6          B7
0500:ABCD 0500:0678 0500:5678 0500:0AAA 0500:0BBB 0000:0000 0000.0000 0000:0000

SRTS>b

    B0          B1          B2          B3          B4          B5          B6          B7
0500:ABCD 0500:0678 0500:5678 0500.0AAA 0500:0BBB 0000:0000 0000:0000 0000:0000
z
```

```
    B0          B1          B2          B3          B4          B5          B6          B7
0000:0000 0000:0000 0000:0000 0000:0000 0000:0000 0000:0000 0000:0000 0000:0000

SRTS>d345,40

segm:offs   0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   ...Ascii-code...
0003:0340                  00 00 00 00 00 00 00 00 00 00 00        ...........
0003:0350   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0003:0360   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0003:0370   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0003.0380   00 00 00 00 00                                    .....

SRTS>d123,50

segm:offs   0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   ...Ascii-code...
0003:0120                  65 20 7A 61 69 6E 73 74 61 6C 6F 77 61     e zainstalowa
0003:0130   6E 65 20 7A 61 64 61 6E 69 65 00 07 1B 59 37 3E   ie zadanie...Y7>
0003:0140   73 74 6F 73 20 6E 69 65 20 75 7A 79 77 61 6E 79   stos nie uzywany
0003:0150   00 07 07 00 73 65 67 6D 3A 6F 66 66 73 20 20 20   ....segm:offs
0003:0160   20 30 20 20 31 20 20 32 20 20 33 20 20 34 20 20    0  1  2  3  4
0003:0170   35 20 20                                        5

SRTS>
```

```
SRTS>f345,ab,40

SRTS>s345

segm:offs   0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   ...Ascii-code...
0003:0340   00 00 00 00 00 AB AB AB AB AB AB AB AB AB AB AB   ................
0003:0350   AB AB AB AB AB AB AB AB AB AB AB AB AB AB AB AB   ................
0003:0360   AB AB AB AB AB AB AB AB AB AB AB AB AB AB AB AB   ................
0003:0370   AB AB AB AB AB AB AB AB AB AB AB AB AB AB AB AB   .... ...........
0003:0380   AB AB AB AB AB AA BB CC DD EE FF 00 00 00 00 00   ................

SRTS>8:c2          244        F4
```

```
244         F4          .
244         F4          .
244         F4          .
244         F4          .
244         F4          .
244         F4          .
244         F4          .

SRTS>kBGT

SRTS>
```

```
SRTS>gBGT

SRTS>1345,ee

SRTS>1345,50,ee

SRTS>13:345,40,ee

SRTS>13:50,ee

SRTS>13:345,50,ee

0003:0389

SRTS>e

R


E0 at address 0500:0592

SRTS>
```

Uwaga: W wyniku zadziałania zadania BGT po komendzie "e" = monitora, na ekranie zostały skasowane linie zawierające ustawienie pułapki B0, samą dyrektywę "e" oraz wypisaną znak "R" (przez BGT).

```
SRTS>nb
0500:0592 55              PUSH    BP

0500:0592   55
0500:05A7 E8C504          CALL    0A6F              ST 0003,0CFA

0500:0AA7 7D08            JGE     0AB1                        FALSE

0500:0AAF EB0C            JMP     SHORT 0ABD

0500:0AC8 E88901          CALL    0C54              ST=14AD,4854,0CF4,05AA,0005

0500:0C76 7517            JNE/JNZ 0C8F                      TRUE

0500:0C8F FFE0            JMP     AX

0500:0AD1 C3              RET                       ST=05AA,0005,0CFA,38B6,0005

SRTS>n
0500:0AD1 C3              RET                       ST=05AA,0005,0CFA,38B6,0005

0500:0AD1 - C3
0500:05AA 83C402          ADD     SP,+02h           ST=0005,0CFA,38B6,0005,0D00
```

```
SRTS>r

 AX   BX   CX   DX   DI   SI   BP   SP   SS   ES   DS   CS   IP   FL
0ACE 14AD 7576 0001 14AD 0E2C 0CF4 0CF2 0003 0500 0003 0500 05AA F046

SRTS>u
 AX   BX   CX   DX   DI   SI   BP   SP   SS   ES   DS   FL =....ODITSZ.A.P.C
0ACE 14AD 7576 0001 14AD 0E2C 0CF4 0CF2 0003 0500 0003 F046 1111000001000110
0500:05AA 83C402        ADD     SP,+02h                 SI=0005,0CFA,38B6,0005,0D00


 AX   BX   CX   DX   DI   SI   BP   SP   SS   ES   DS   CS   IP   FL
0ACE 14AD 7576 0001 14AD 0E2C 0CF4 0CF2 0003 0500 0003 0500 05AA F046
0500:05AA - 83
 AX   BX   CX   DX   DI   SI   BP   SP   SS   ES   DS   FL =....ODITSZ.A.P.C
0ACE 14AD 7576 0001 14AD 0E2C 0CF4 0CF4 0003 0500 0003 F102 1111000100000010
0500:05AD 833E4E1400    CMP     WORD PTR [144E],+00h            DS:144E=0000

 AX   BX   CX   DX   DI   SI   BP   SP   SS   ES   DS   FL =....ODITSZ.A.P.C
0ACE 14AD 7576 0001 14AD 0E2C 0CF4 0CF4 0003 0500 0003 F146 1111000101000110
0500:05B2 740F          JE/JZ   05C3                    TRUE

SRTS>
```

```
SRTS>pBGT

segm:offs    0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    ...Ascii-code...
0003:0CF0                  FA 0C B6 38 05 00 00 01 1B 09 00 00      ...8........
0003:0D00   14 0D 0B 3A FC 36 00 00 3E 1A 3D 17 04 00 D8 00   ...:.6..>.=. ...
0003:0D10   7D 14 BD 14 00 3A 24 CD B5 47 00 05 97 F2 00 00   }....:$..G......
0003:0D20   6B 08 C1 00 2C 0D B4 46 18 19 CF 4F 2C 0D         k...,..F...O,.

SRTS>t

lp.  zad.    stan     adwe  pwst  bwsb  asbc   ltakt  cykl  l.cyk  sch  sem  acz
 0   SYS     READY    0212  132E  1306  0000       0     0      0    0
 1   KBD     SUSP     3C05  122E  121B  150C      -1     0      0    1    0    1
 2   MON     SUSP     3E1B  112E  1114  150B      14     0      0    1    0    1
 3   PRT     SUSPE    3F5F  102E  101C  0574      -1     0      0    1
 4   MMC     SUSPE    3FB7  0F2E  0F0A  0554      -1     0      0    1
 5   ADS     READY    4854  0E2E  0F2A  0000      -1     0      0    1
 6   BGT     RUNNING  4686  0D2E  0CFA  0000      -1     0      0    0

SRTS>5b2/16     = 004E r 0006

SRTS>5b2+ab     = 065D


SRTS>u5b2

0500:05B2 740F          JE/JZ   05C3                            t.

SRTS>u592

0500:0592 55            PUSH    BP                              U

SRTS>20u592,b

0500:0597 E8C504        CALL    0A6F                            ..

SRTS>5uCa7

0500:05A7 E8C504        CALL    0A6F                            ...
0500:05AA 83C402        ADD     SP,+02h                         ...

SRTS u                          brak parametru

SRTS>
```


17

5. Zestawienie komend monitora operatorskiego SRTS

Poniższa tabela podaje listę dyrektyw monitora operatorskiego SRTS systemu operacyjnego czasu rzeczywistego SIRTOS w kolejności alfabetycznej. Parametr opcjonalny dyrektywy podano w nawiasie kwadratowym, brak parametru oznaczono znakiem "-".

Licznik powtórzeń	Nazwa komendy	Arg. 1	Arg. 2	Arg. 3
	B	-	-	-
	C	[seg1:]off1	[,seg2]:off2	[,count]
	D	[segm:]offs	[,count]	-
	E	-	-	-
	F	[segm:]offs	,arg	[,count]
	G	task	-	-
[count]	I	addr	-	-
[count]	J	addr	-	-
	K	task	-	-
	L	[segm:]offs	[,count]	,arg
	M	-	-	-
	MB	-	-	-
	MT	-	-	-
	N	-	-	-
	NB	-	-	-
	NT	-	-	-
[count]	O	addr	,arg	-
	P	task	-	-
	R	-	-	-
	S	[segm:]offs	-	-
	T	-	-	-
[count]	U	[segm:]offs	[,B]	-

Uwaga: W zestawieniu umieszczono w linii komendy spacje ze względu na czytelność zapisu; podczas używania dyrektyw spacje te nie są dozwolone.

Komendy pomocnicze

CR (karetka)

arg1	+	arg2
arg1	-	arg2
arg1	*	arg2
arg1	/	arg2