

Zarządzanie zadaniami w systemach embedded

Super Simple Tasker, część 3

Dla tych, którzy po raz pierwszy zetknęli się z systemami operacyjnymi typu embedded, jak na przykład μ C/OS czy eCos, sposób ich działania może wydać się skomplikowany, sztuczny i mało intuicyjny. W miarę nabierania doświadczenia odczucie to wprawdzie słabnie, nadal pozostaje jednak pewna nieufność. W niniejszym artykule przedstawiamy prostsze rozwiązania alternatywne pozwalające zarządzać zadaniami w systemie embedded.

Kontynuujemy implementację SST w języku C. Do tego, by Super Simple Tasker mógł rozpocząć pracę, niezbędne jest zdefiniowanie jeszcze kilku funkcji.

W SST zdarzenia są przekazywane za pomocą jednej funkcji SST_PostEvent, której implementację przedstawiono na **list. 9**. Komentarz do odnośników z tego listingu jest następujący:

- 1. Czy identyfikator zadania, do którego należy przekazać komunikat, jest poprawny?
- 2. Jeżeli nie, to nic nie rób i wyjdź z funkcji, zgłaszając błąd.
- 3. Zablokuj przerwania.
- 4. Dodaj komunikat do kolejki komunikatów przypisanej do zadania o identyfikatorze *id*.
- 5. Jeżeli operacja przekazania komunikatu udała się (w kolejce było wolne miejsce)...
- 6. to sprawdź, czy przekazany komunikat jest pierwszy w kolejce...

List. 9. Implementacja funkcji SST_PostEvent

```
bool SST_PostEvent(SST_TaskID_T id, SST_Signal_T sig, SST_Param_T par)
{
    SST_DECLARE_INT_USAGE;
    bool retval;
    CPU_Base_T no_events;
    if (!SST_IS_VALID_TASK(id))
    {
        return false;
    }
    SST_INT_LOCK_AND_SAVE();
    no_events = put_event(&_SST_tasks[id].queue, sig, par);
    if (no_events)
    {
        if ((CPU_Base_T) 1 == no_events)
        {
            _SST_Add_As_Ready(id);
        }
        _SST_Schedule(SST_INT_STATE);
    }
    SST_INT_LOCK_RESTORE();
    return (no_events != 0);
}
```

- 7. Jeżeli tak, to dodaj zadanie (dla którego przekazano właśnie komunikat) do listy zadań gotowych do wykonania (wykonanie tej funkcji jest stosunkowo kosztowne i warto ograniczyć liczbę jej wywołań – o ile to możliwe)...
- 8. i uruchom scheduler.
- 9. Odblokuj przerwania.
- 10. Zwróć informację, czy operacja przekazania komunikatu się udała.

W końcu przyszła pora na „serce” systemu, czyli zarządcę zadań – funkcję _SST_Schedule. Przedstawiono ją na **list. 10**, a komentarz do odnośników jest następujący:

- 1. Funkcja jest zawsze wywoływana z zablokowanymi przerwaniami.
- 2. Jeżeli aktualnie wykonywany jest kod przerwania lub nie ma zadań oczekujących na wykonanie, wtedy nic nie rób (zakończ funkcję).
- 3. Zapamiętaj priorytet aktualnie wykonywanego zadania.
- 4. Rozpocznij pętlę przetwarzania zadań z listy gotowych do wykonania.
- 5. Pobierz wskaźnik do struktury zawierającej informację o zadaniu przeznaczonym do wykonania (o najwyższym priorytecie z listy zadań oczekujących na wykonanie).
- 6. Jeżeli priorytet zadania gotowego do wykonania jest wyższy od priorytetu zadania, z którego była wywołana funkcja _SST_Schedule...
- 7. to przerwij wykonywanie pętli przetwarzania zadań.
- 8. Jeżeli kolejka komunikatów związana z zadaniem przeznaczonym do wykonania nie jest pusta...

- 9. Jeżeli był to ostatni komunikat w kolejce danego zadania...
- 10. to usuń to zadanie z listy zadań gotowych do wykonania.
- 11. Ustaw aktualny priorytet na wartość priorytetu zadania wybranego do wykonania.
- 12. Odblokuj przerwania.
- 13. Wykonaj zadanie.
- 14. Zablokuj przerwania.
- 15. Jeżeli kolejka komunikatów dla zadania przeznaczonego do uruchomienia jest pusta...
- 16. to usuń to zadanie z listy zadań gotowych do wykonania (w normalnych warunkach nie powinna

List. 10. Główna funkcja systemu _SST_Schedule

```
void _SST_Schedule(SST_DECLARE_INT_USAGE)
{
    CPU_Base_T cprio;
    volatile TaskCB_T *tcb;
    SST_Event_T e;

    if ((SST_CurrPrio >= SST_MIN_INT_PRIO) || !SST_IS_VALID_TASK(_SST_first))
    {
        return;
    }
    cprio = _SST_CurrPrio;
    do
    {
        tcb = (TaskCB_T *) &sst_tasks[hp_task];
        if (tcb->prio <= cprio)
        {
            break;
        }
        if (get_event(&tcb->queue, &e))
        {
            if (0 == tcb->queue.used)
            {
                Remove_First_Ready_Task(tcb);
                SST_CurrPrio = tcb->prio;
                SST_INT_LOCK_RESTORE();
                tcb->task(e);
                SST_INT_LOCK();
            }
            else
            {
                Remove_First_Ready_Task(tcb);
            }
        } while (SST_IS_VALID_TASK(hp_task));
        _SST_CurrPrio = cprio;
    }
}
```

List. 11. Zestaw definicji, które wymagają specyficznej implementacji zależnej od użytego procesora i kontrolera przerwań

```
CPU_Base_T
CPU_SR_T
SST_Start_Nested_Ints()
SST_Stop_Nested_Ints()
SST_disable_ints_save(_old_)
SST_disable_ints()
SST_enable_ints()
SST_restore_ints(_old_)
```



Qwerty®
zaufaj nam

PROJEKTUJEMY
PRODUKUJEMY
SPRZEDAJEMY

sprawdź naszą
nową stronę!
www.qwerty.pl

- specjalizujemy się w projektowaniu i produkcji klawiatur, elewacji, tabliczek i zestyków foliowych
- wykwalfikowani pracownicy pomogą dopasować odpowiednią technologię do Państwa wymagań a wysokiej jakości materiały i nowoczesne technologie zagwarantują niezawodność naszych wyrobów

www.qwerty.pl

PRODUCENT KLAWIATUR FOLIOWYCH



Towarzystwo Elektrotechnologiczne Qwerty Sp. z o.o.
ul. Siewna 21, 94-250 Łódź, e-mail qwerty@qwerty.pl
tel. (42)632-47-92, 633-32-84, 630-42-64, fax (42)632-85-93



USŁUGI MONTAŻU PŁYTEK ELEKTRONICZNYCH

- ➔ Oferujemy usługi montażu przewlekane i powierzchniowe
- ➔ Uruchamianie wyrobów
- ➔ Dostawa podzespołów
- ➔ Wykonujemy szablony z blachy stalowej



WSZYSTKO DO MONTAŻU ELEKTRONICZNEGO



- ➔ Pasty lutownicze
- ➔ Topniki,
- ➔ Preparaty myjące
- ➔ Lakiery, żywice
- ➔ Narzędzia
- ➔ Stacje lutownicze
- ➔ Igły testowe
- ➔ Antystatyka

www.FERYSTER.pl

POWER INTEGRATIONS

FERYSTER Sp. z o.o. z siedzibą w Łodzi
PRODUCENT ELEMENTÓW INDUKCYJNYCH



**KLUCZE PI
(TNY III I PKS)
JUŻ NA NASZYM
MAGAZYNIE!**

WWW.FERYSTER.PL

RFID
TRANSPONDERY
STEROWNIKI
CZYTNIKI

www.mikrokontrola.pl
mikrokontrola

ul. Wólczyńska 55, 01-908 Warszawa, tel.: 0-22/865 55 43
fax: 0-22/ 865 55 44, e-mail: biuro@mikrokontrola.pl



ul. Zwoleńska 43/43A,
04-761 Warszawa
tel. (22) 615 73 71, 615 64 31,
fax. (22) 615 73 75
info@semicon.com.pl,
www.semicon.com.pl

Wiedza

- **LabTool-T400** – superszybki programator pamięci FLASH. Firma Advantech Equipment Corporation posiada w swojej ofercie programator przeznaczony do programowania pamięci Flash o napięciu zasilania i poziomach napięć wejściowych/wyjściowych od 1,2 V do 3,3 V. Programator obsługuje pamięci 8, 16 i 32 o maksymalnej pojemności do 128 GB.
- **Zasilacze DIN.** Zasilacze Mean Well serii DR przeznaczone do systemów automatyki przemysłowej, energetyki, zabezpieczeń przemysłowych a także systemów security. Wyróżnia je montaż na szynie DIN o szerokości 35 mm (TS-35mm).
- **Firma National Instrument wprowadza na rynek.** Firma National Instruments rozszerza swoją linię cyfrowych multimetrów (DMM) o nowe, tanie 6½-cyfrowe multimetry przeznaczone dla magistrali PCI Express oraz PCI.
- **Układy Samoczynnego Załączenia Rezerwy – SZR.** Wymagania związane z niezawodnością i ciągłością zasilania wielu obiektów są coraz większe, dlatego odbiorcy często decydują się na zastosowanie układów eliminujących dłuższe przerwy w zasilaniu. Jednym z rozwiązań są układy Samoczynnego Załączenia Rezerwy SZR.
- **ioLogikR2140** – moduł wejść/wyjść analogowych z interfejsem RS-485. R2140 to najnowszy moduł pomiarowy z serii ioLogik2000. Jest to moduł wyposażony w 8 analogowych wejść i 2 analogowe wyjścia. Komunikacja z komputerem PC odbywa się poprzez interfejs RS-485.

Partnerzy



Aktualności

- **MOXA EM-1240 – PRZEMYSŁOWY KOMPUTER JEDNOPŁYTKOWY DO ZASTOSOWAŃ WBUDOWANYCH.** EM-1240 to przemysłowy komputer jednoplaskowy przeznaczony do instalacji w aplikacjach wbudowanych. Małe rozmiary modułu umożliwiają łatwą integrację z innymi urządzeniami, natomiast szeroka funkcjonalność pozwala na elastyczne dopasowanie aplikacji do wymaganego projektu.
- **WYJĄTKOWO MAŁY I NIEDROGI KOMPUTER PRZEMYSŁOWY.** Polskie biuro firmy Axiomtek, tajwańskiego producenta komputerów przemysłowych ma przyjemność zaprezentować miniaturowy komputer kompaktowy eBOX646 o rozmiarach: 200mm szer. x 150 mm gł. x 44 mm wys. Szczególną zaletą komputera eBOX646 jest zdolność do pracy w rozszerzonym zakresie temperatur: -10°C – 50°C!
- **V PANEL EXPRESS – NAJWYDAJNIEJSZY KOMPUTER PANELOWY NA RYNKU.** V Panel Express to obecnie najwydajniejszy komputer panelowy dostępny na rynku. Bardzo solidne-niemieckie wykonanie, niski pobór mocy, odporność na szoki i wibracje, szeroki zakres temperatur pracy, różnorodność konfiguracji oraz niemal że nieograniczone możliwości rozbudowy dedykują go do wszelkich zastosowań wizualizacyjno-sterujących.
- **MOXA EDS-516A – 16 PORTOWY ZARZĄDZALNY, REDUNDANTNY SWITCH PRZEMYSŁOWY.** MOXA EDS-516A to nowa seria zarządzalnych, przemysłowych switchy Ethernetowych, znanego producenta urządzeń do sieci przemysłowych – firmy Moxa.
- **FIRMA NATIONAL INSTRUMENTS ODNOSI KOLEJNE SUKCESY ZE WSCHODNIOEUROPEJSKIMI INTEGRATORAMI.** Firma National Instruments (Nasdaq: NATI) otworzyła pierwsze biuro sprzedaży w Europie Wschodniej w 1999 r. Od tego czasu notuje nieustanny rozwój i rozszerza swoją działalność w tym szybko rozwijającym się regionie.
- **PANELPC Z WYŚWIETLACZEM 12,1" (FRONT ZE STALI NIERDZEWNEJ, BEZ LOGA) ZA 1 499 EURO.** KONTRON wprowadza na rynek specjalną wersję bardzo popularnego komputera panelowego z serii „ECO PANEL” posiadającą front ze stali nierdzewnej IP65 co tym samym dedykuje go głównie do wykorzystania w przemyśle spożywczym ale również do wizualizacji procesów przemysłowych oraz innych aplikacji tego typu.

Redakcja

sterowniki.pl Sp. z o.o.
tel. 022 499 88 39
www.sterowniki.pl
e-mail: sterowniki@sterowniki.pl

się zdarzyć sytuacja, że zadanie jest gotowe do wykonania i nie ma komunikatów).

17. Wykonuj pętlę przetwarzania zadań dopóki istnieją zadania gotowe do wykonania.
18. Odtwórz priorytet zadania, z którego była wywołana funkcja `SST_Schedule`.

W celu ułatwienia implementacji SST na różnych mikrokontrolerach (portowania) wydzielony został zestaw tych definicji, które wymagają specyficznej implementacji, zależnej od użytego procesora i kontrolera przerwań. Są to typy danych i makra przedstawione na list. 11. Poszczególne deklaracje oznaczają:

1. „Naturalny” typ danych dla danego mikrokontrolera; operacje na tym typie powinny być z definicji najbardziej efektywne.
2. Typ danych pozwalający na przechowanie informacji o tym, czy przerwania są włączone lub wyłączone; ko-rzystają z niego makra `SST_disable_ints_save` i `SST_restore_ints`.
3. Wykonuje operacje niezbędne dla danego mikrokontrolera w celu umożliwienia zagnieżdżenia przerwań (w tym ich odblokowania).
4. Blokuje możliwość zagnieżdżenia przerwań (i same przerwania).
5. Zapamiętuje w zmiennej `_old_status` (włączone/wyłączone) przerwań i je wyłącza.
6. Wyłącza przerwania.
7. Włącza przerwania.
8. Odtwarza stan przerwań zapamiętany w zmiennej `_old_`.

Implementacja specyficzna dla AVR

W tym przypadku implementacja jest bardzo prosta ze względu na brak trybu pracy CPU spe-

cyficznego dla przerwań oraz brak rozbudowanego kontrolera przerwań. Definicje i makra specyficzne dla AVR-a zostały przedstawione na list. 12.

Implementacja specyficzna dla mikrokontrolera z rodziny LPC2xxx

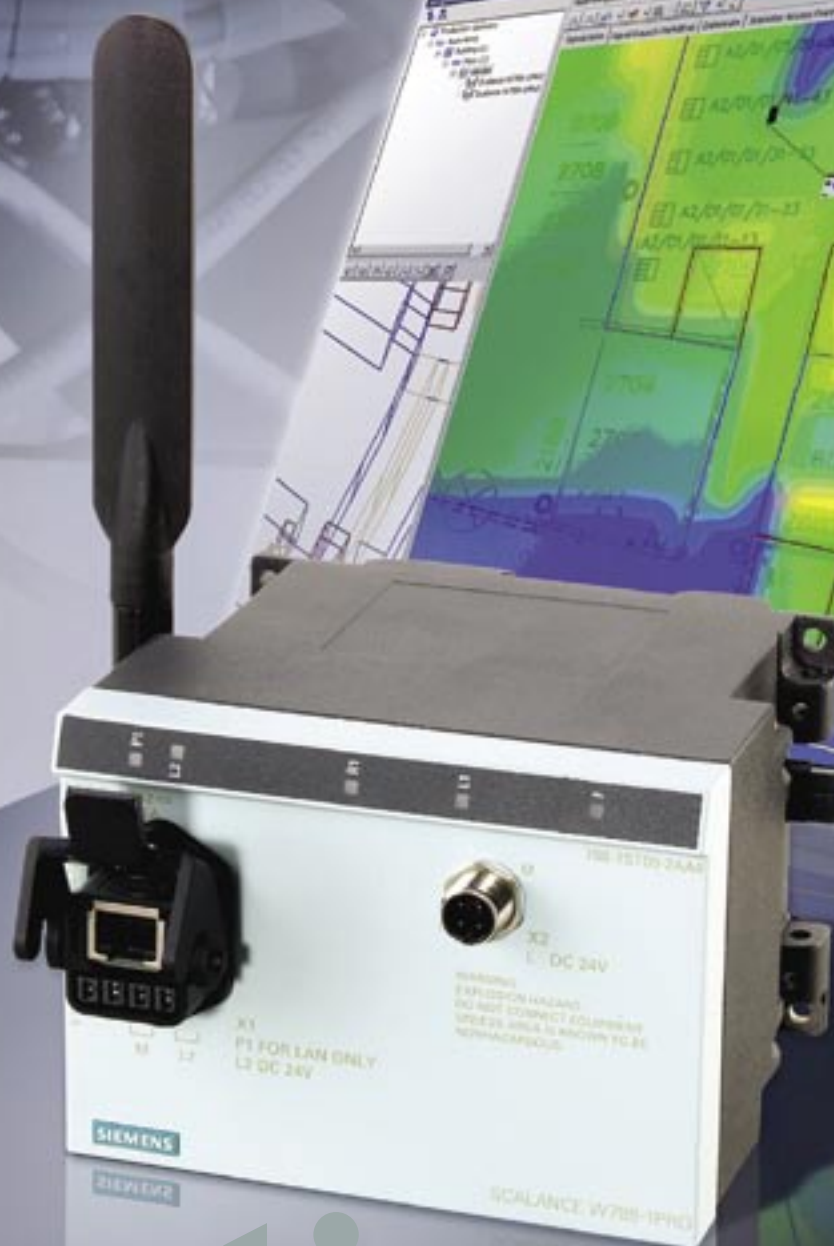
W tym przypadku sprawa się nieco komplikuje, a to z powodu wbudowanego kontrolera przerwań (VIC) i specyfiki obsługi przerwań w procesorze ARM. Zgodnie z zaleceniami firmy ARM, jak i Philips (nota aplikacyjna AN10381), obsługa przerwań zagnieżdżonych wymaga kilku dodatkowych operacji:

- zapamiętania (na stosie) rejestru `SPSR_irq`,
- przełączenia się do modu pracy sys (system) i włączenia przerwań,
- zapamiętania (na stosie) rejestru `lr` (adres powrotu z funkcji).

Przy blokowaniu zagnieżdżenia przerwań należy wykonać powyższe działania w odwrotnej kolejności. Deklaracje dla mikrokontrolera ARM przedstawiono na list. 13. W przypadku stosowania mikrokontrolerów (opartych na tym samym rdzeniu) innych producentów, np. AT91SAM7, różnica implementacji będzie dotyczyła tylko sposobu obsługi kontrolera przerwań.

Jeśli planowane jest używanie trybu THUMB, należy pamiętać, że nie ma w nim bezpośredniego dostępu do rejestru statusu (CPSR), więc makra powinny zawierać na początku dodatkowy kod przełączający procesor do trybu ARM, a na końcu wracający do trybu THUMB. Alternatywnym (polecanym) rozwiązaniem jest użycie przerwania programowego do włączania i wyłączania przerwań. Makra użyte do zarządzania przerwaniami dla trybu

SCALANCE W – technologia radiowa w sieciach przemysłowych



simatic net SCALANCE W

Nieograniczony dostęp do informacji niezależnie od miejsca pracy. Moduły SCALANCE W zapewniają wydajną, bezpieczną i stabilną komunikację w warunkach przemysłowych. Praca w standardzie IEEE 802.11 b/g lub 802.11a gwarantuje niezawodną transmisję z prędkością do 54 MB/s.

SIEMENS

Branża Automation and Drives w Polsce

**Siemens Sp. z o.o.
Automation & Drives**
ul. Żupnicza 11
03-821 Warszawa
tel. 022 870 9862
fax 022 870 9868

**Biuro regionalne
w Gdańsku**
ul. Grunwaldzka 413
80-309 Gdańsk
tel. 058 764 6092
fax 058 764 6099

**Biuro regionalne
w Katowicach**
ul. Gawronów 22
40-527 Katowice
tel. 032 208 4134
fax 032 208 4139

**Biuro regionalne
w Krakowie**
ul. Kraszewskiego 36
30-110 Kraków
tel. 012 422 7780
fax 012 427 2629

**Biuro regionalne
w Poznaniu**
ul. Ziębicka 35
60-164 Poznań
tel. 061 664 9861
fax 061 664 9864

**Biuro regionalne
we Wrocławiu**
ul. Ostrowskiego 30
53-238 Wrocław
tel. 071 777 5060
fax 071 777 5050

www.siemens.pl/simatic

e-mail: simatic.pl@siemens.com

szkolenia.pl@siemens.com

List. 12. Definicje i makra specyficzne dla mikrokontrolera AVR

```

typedef uint8_t CPU_Base_T;
typedef uint8_t CPU_SR_T;
#define SST_Start_Nested_Ints()
sei()
#define SST_Stop_Nested_Ints()
do { } while (0)
#define SST_disable_ints_save(_old_)
{
do {
\
(_old_) = SREG & _BV(SREG_
I); \
cli();
\
} while (0)
#define SST_disable_ints()
cli()
#define SST_enable_ints()
sei()
#define SST_restore_ints(_old_)
do { SREG |= (_old_); } while(0)

```

ARM przedstawiono na list. 14.

Zużycie pamięci RAM i wielkość kodu dla prezentowanego rozwiązania

Zapotrzebowanie na pamięć RAM i Flash w zależności od liczby zadań i długości kolejki zestawiono w tab. 1.

Mam nadzieję, że przedstawione rozwiązanie pozwoli przynajmniej niektórym Czytelnikom na wyrwanie się z „zakłętą kłęb” zarządzania typem *Cyclic Scheduling* (nieskończona pętla z funkcjami/zadaniami wywoływany sekwencyjnie), natomiast Czytelnicy używający bardziej zaawansowanych rozwiązań uzyskają nowe narzędzie, przydatne tam, gdzie zasoby mikrokontrolera stanowią ograniczenie dla klasycznego RTOS-a.

Pełny kod wraz z dokumentacją

List. 13. Definicje i makra specyficzne dla mikrokontrolera ARM

```

typedef uint32_t CPU_Base_T;
typedef uint32_t CPU_SR_T;
#define SST_Start_Nested_Ints()
do { VICVectAddr = 0xFF; ENABLE_
NEST_INTS(); } while (0)
#define SST_Stop_Nested_Ints()
DISABLE_NEST_INTS()
#define SST_disable_ints_save(_old_)
DISABLE_INTS_SAVE(_old_)
#define SST_disable_ints()
DISABLE_INTS()
#define SST_enable_ints()
ENABLE_INTS()
#define SST_restore_ints(_old_)
RESTORE_INTS(_old_)

```

Tab. 1.

	ARM	AVR
Część zależna od liczby zadań i maksymalnej długości kolejki dla 5 zadań i kolejki o długości 4	288 bajtów (RAM)	77 bajtów (RAM)
Część niezależna od liczby zadań (<i>scheduler</i>)	828 bajty (Flash)	620 bajty (Flash)
Część niezależna od liczby zadań (implementacja <i>mutex-a</i>)	148 bajtów (Flash)	72 bajty (Flash)

zamieszczamy na płycie CD-EP4/2007B. Materiały źródłowe można też znaleźć pod adresem: <http://tech.groups.yahoo.com/group/lpc2000/files>, http://www.avrfreaks.net/index.php?module=Freaks%20Academy&func=viewItem&item_type=project&item_id=725. Strony wymagają logowania.

Artur Lipowski
LAL@pro.onet.pl

List. 14. Makra użyte do zarządzania przerwaniami dla trybu ARM

```

#define ENABLE_NEST_INTS()
asm volatile (
"mrs lr, SPSR \n\t"
"stmfd sp!, {lr} \n\t"
"msr CPSR_c, #0x1F \n\t"
"stmfd sp!, {lr} \n\t"
)

#define DISABLE_NEST_INTS()
asm volatile (
"ldmfd sp!, {lr} \n\t"
"msr CPSR_c, #0x92 \n\t"
"ldmfd sp!, {lr} \n\t"
"msr SPSR_cxsf, lr \n\t"
)

#define DISABLE_INTS_SAVE(_old_)
asm volatile (
"mrs %0,CPSR \n\t"
"mrs r4,CPSR \n\t"
"orr r4,r4,#0x80 \n\t"
"msr CPSR_c,r4 \n\t"
: "=r"(_old_)
:
: "r4"
);

#define DISABLE_INTS()
asm volatile (
"mrs r4,CPSR \n\t"
"orr r4,r4,#0x80 \n\t"
"msr CPSR_c,r4 \n\t"
:
:
: "r4"
);

#define ENABLE_INTS()
asm volatile (
"mrs r3,CPSR \n\t"
"bic r3,r3,#0x80 \n\t"
"msr CPSR_c,r3 \n\t"
:
:
: "r3"
);

#define RESTORE_INTS(_old_)
asm volatile (
"mrs r3,CPSR \n\t"
"and r4,%0,#0x80 \n\t"
"bic r3,r3,#0x80 \n\t"
"orr r3,r3,r4 \n\t"
"msr CPSR_c,r3 \n\t"
:
: "r"(_old_)
: "r3", "r4"
);

```



G2RV

Advanced Industrial Automation

PRZekaźnikowy moduł interfejsowy G2RV – JAKOŚĆ ZA ROZSĄDNĄ CENĘ

Bardzo wąska (6 mm) obudowa z mocnymi wyprowadzeniami zapobiegającymi wygięciom nóżek przełącznika podczas serwisowania i zwiększającymi powierzchnię styku z podstawką.

Wbudowane wskaźniki dołączenia zasilania (LED) i wskaźnik mechaniczny przełączania styków, nie występujący w przełącznikach innych producentów upraszcza uruchamianie i serwisowanie.

- Przezroczysta obudowa ułatwia diagnozowanie ewentualnych awarii.
- Dopuszczalne obciążenie: 6 A dla 250 VAC i 30 VDC
- Wytrzymałość mechaniczna: 5 000 000 przełączeń
- Wytrzymałość elektryczna: 100 000 operacji

OMRON ELECTRONICS Sp. z o.o.
ul. Mariana Sengera „Cichego” 1
02-790 Warszawa

Tel.: 0 (prefix) 22 645 78 60
Fax: 0 (prefix) 22 645 78 63
www.omron.com.pl

