




Traffic  
System Development  
Consultancy

**MV-MODULE  
RWSC  
versie 3**

## Inhoudsopgave

<b>1 Inleiding.....</b>	<b>3</b>
<b>2 Namen in applicatie.....</b>	<b>4</b>
2.1 Opslag structuur.....	4
2.2 Gedefinieerde structuren.....	4
<b>3 Signaalgroep informatie bij bijzondere ingrepen.....</b>	<b>6</b>
<b>4 Extra's.....</b>	<b>7</b>

	Datum	Versie	Status	Documentnaam	Auteur
	02-11-06	001		MV Module RWSC versie 3.0	RKA

## 1 Inleiding

Om enkele problemen met versie 1 en 2 van de mv\_rwsc module op te lossen, is het noodzakelijk een nieuwe versie te ontwikkelen.

De volgende problemen dienen in deze nieuwe versie opgelost te worden:

1. Mogelijkheid tot creëren van meerdere MV bestanden in een FLASH simulatie met meerdere regelingen.
2. Het opnemen van koppelsignalen in het log proces.
3. Naamgeving van bestanden dienen aan 8.3 conventie te voldoen wegens compilatie problemen bij fabrikanten.
4. In en uitgangen op een zekere manier loggen, onafhankelijk van extern (buiten de regeling) aangeboden informatie betreffende bestaande in- en uitgangen.
5. Ondersteuning signaalgroep informatie bij bijzondere ingrepen

## 2 Namen in applicatie

Ten behoeve van ondersteuning van meerdere mv log bestanden binnen 1 FLASH simulatie is het noodzakelijk niet meer afhankelijk te zijn van een extern bestand (mvf.dat), aangezien FLASH altijd het bestand uit dezelfde directory leest, en derhalve maar de gegevens van 1 enkele regeling op de vastgelegde manier kan verwerken.

Een oplossing zou kunnen zijn om een mvf.dat bestand zodanig te construeren dat meerdere regelingen in 1 bestand geplaatst zou kunnen worden, maar dan dien je eventueel meerdere mvf.dat bestanden per regeling aan te maken, plus het kopiëren van het mvf.dat bestand naar de FLASH directory, dat nu telkens dient te gebeuren, blijft bestaan.

De gekozen oplossing is dehalve dat de namen voor alle I/O in de regeling zelf wordt opgenomen. Dit levert de volgende voordelen op:

1. gegarandeerd dezelfde benamingen in simulatie en regelaar
2. niet langer afhankelijk van implementatie van fabrikanten
3. ondersteuning mogelijk van meerdere regelingen in 1 FLASH simulatie
4. het kopiëren van mvf.dat naar FLASH simulatie directory is niet langer noodzakelijk

### 2.1 Opslag structuur

De benodigde structuur om I/O namen op te slaan bestaat uit een combinatie van een source nummer (nummer of 'naam' uit crapdef.h) en een karakter string (de te gebruiken naam):

```
typedef struct kwc_info_type {  
    int    source;  
    char*  name;  
};
```

Daarnaast dient de naam van de regeling vastgelegd te worden in een karakter string.

### 2.2 Gedefinieerde structuren

In het bestand mv\_rwsc.h worden de volgende externe structuren gedefinieerd om de namen vast te leggen:

```
extern char*          kwc_info_system;  
extern struct kwc_info_type kwc_info_sg[];  
extern struct kwc_info_type kwc_info_dp[];  
extern struct kwc_info_type kwc_info_uit[];  
extern struct kwc_info_type kwc_info_in[];  
extern struct kwc_info_type kwc_info_koppel_uit[];  
extern struct kwc_info_type kwc_info_koppel_in[];
```

De benodigde data dient aan de regeling te worden toegevoegd.

Voorbeeld:

```
char* kwc_info_system = "VR1xxx";  
  
struct kwc_info_type kwc_info_sg[] = {  
    { SG02      , "SG02"      },  
    { SG03      , "SG03"      },
```

```

    { SG05          , "SG05"          },
};

struct kwc_info_type kwc_info_dp[] = {
    { SG02_1      , "D02.1"          },
    { SG02_2      , "D02.2"          },
    { SG03_1      , "D03.1"          },
};

struct kwc_info_type kwc_info_uit[] = {
    { 0           , "BLOK"           }, /* altijd opnemen */
    { 1           , "filemelding"   },
};

struct kwc_info_type kwc_info_in[] = {
    { 0           , "Fixatie"       }, /* altijd opnemen */
};

struct kwc_info_type kwc_info_koppel_uit[] = {
    { 0           , ""              },
};


struct kwc_info_type kwc_info_koppel_in[] = {
    { 0           , ""              },
};

int kwc_nr_sg      = sizeof(kwc_info_sg) / sizeof(kwc_info_sg[0]);
int kwc_nr_uit     = sizeof(kwc_info_uit) / sizeof(kwc_info_uit[0]);
int kwc_nr_dp      = sizeof(kwc_info_dp) / sizeof(kwc_info_dp[0]);
int kwc_nr_in      = sizeof(kwc_info_in) / sizeof(kwc_info_in[0]);
int kwc_nr_kop_in  = 0; /* sizeof(kwc_info_koppel_in) / sizeof(kwc_info_koppel_in[0]); */
int kwc_nr_kop_uit = 0; /* sizeof(kwc_info_koppel_uit) / sizeof(kwc_info_koppel_uit[0]); */

```

### Opmerkingen:

- Indien een regeling geen koppelsignalen bevat, dient het aantal koppelsignalen worden opgegeven door bovenstaand voorbeeld. Indien er wel koppelsignalen zijn, dient het aantal te worden opgegeven door de 'sizeof' constructie die in het voorbeeld hierboven in commentaar staat.
- De eerste uitgang (CIF\_GUS[SGMAXMAX]) is gereserveerd voor blok informatie (ook bij langstwachende structuur)
- De eerste ingang (CIF\_IS[DETMAXMAX]) is gereserveerd voor fixatie.
- Het source nummer van de koppelsignalen is het gebruikte hulpfunctie nummer uit de KOPPEL\_tabel.
- Gaten in source nummering zijn toegestaan.

	Datum	Versie	Status	Documentnaam	Auteur
	02-11-06	001		MV Module RWSC versie 3.0	RKA

### 3 Extra's

Om uitgang 'blok' een zinnige betekenis te geven in het uitgangen overzicht van de fasenlog weergave in de Kwaliteitscentrale (die geen numerieke weergave geeft, enkel een op/af indicatie), wordt er aan het begin van blok 1 een event 'start uitgangssignaal' gegenereerd, en aan het eind van blok 1 een event 'einde uitgangssignaal'. In de uitgangenweergave bij de fasenlog in de Kwaliteitscentrale zal derhalve bij actief zijn van blok 1 een weergave 'uitgang hoog' worden getoond.