1. Funkcje matematyczne dostępne w sterownikach PLC

(Materiał zaczerpnięty z pracy dyplomowej T. Janaszek)

Sterowniki PLC umożliwiają realizację podstawowych i zaawansowanych operacji matematycznych. Operacje matematyczne, dzieli się na stałoprzecinkowe (realizowane za pomocą liczb naturalnych i całkowitych) i zmiennoprzecinkowe (realizowane za pomocą liczb rzeczywistych). Zarówno liczby stałoprzecinkowe jak i zmiennoprzecinkowe mogą być ze znakiem (signed) lub bez (unsigned). Do realizacji niniejszej pracy wykorzystuje się sterowniki PLC firmy Omron. Dlatego zasadnicza uwaga zostanie zwrócona na operacje matematyczne, które są realizowane za pomocą sterowników kompaktowych i modułowych firmy Omron.

1.1. Operacje stałoprzecinkowe

Sterowniki PLC umożliwiają realizację operacji matematycznych (stałoprzecinkowych) przedstawionych w tabeli 1 [1].

Tabela 1.1. Operacje matematyczne stałoprzecinkowe [1]

Symbol funkcji	Kod funkcji	Nazwa funkcji
+	(400)	Dodawanie liczb 16b bez przeniesienia
+L	(401)	Dodawanie liczb 32b bez przeniesienia
+C	(402)	Dodawanie liczb 16b binarnych z przeniesieniem
+CL	(403)	Dodawanie liczb 32b binarnych z przeniesieniem
+B	(404)	Dodawanie liczb 16b BCD bez przeniesienia
+BL	(405)	Dodawanie liczb 32b BCD bez przeniesienia
+BC	(406)	Dodawanie liczb 16b BCD z przeniesieniem
+BCL	(407)	Dodawanie liczb 32b BCD z przeniesieniem
-	(410)	Odejmowanie liczb 16b bez przeniesienia
-L	(411)	Odejmowanie liczb 32b bez przeniesienia
-C	(412)	Odejmowanie liczb 16b binarnych z przeniesieniem
-CL	(413)	Odejmowanie liczb 32b binarnych z przeniesieniem
-B	(414)	Odejmowanie liczb 16b BCD bez przeniesienia
-BL	(415)	Odejmowanie liczb 32b BCD bez przeniesienia
-BC	(416)	Odejmowanie liczb 16b BCD z przeniesieniem
-BCL	(417)	Odejmowanie liczb 32b BCD z przeniesieniem
*	(420)	Mnożenie liczb 16b ze znakiem
*L	(421)	Mnożenie liczb 32b ze znakiem
*U	(422)	Mnożenie liczb 16b bez znaku
*UL	(423)	Mnożenie liczb 32b bez znaku
Symbol funkcji	Kod funkcji	Nazwa funkcji
*B	(424)	Mnożenie liczb 16b BCD
*BL	(425)	Mnożenie liczb 32b BCD
/	(430)	Dzielenie liczb 16b ze znakiem
/L	(431)	Dzielenie liczb 32b ze znakiem
/U	(432)	Dzielenie liczb 16b bez znaku
/UL	(433)	Dzielenie liczb 32b bez znaku
/B	(434)	Dzielenie liczb 16b BCD
/BL	(435)	Dzielenie liczb 32b BCD

Poniżej zostaną omówione funkcje matematyczne najczęściej wykorzystywane w obliczeniach.

Dodawanie liczb 16b bez przeniesienia

Funkcja +(400) realizuje dodawanie liczb szesnastobitowych hexadecymalnych z zakresu #0000 do #FFFF. Funkcja ta zawiera 3 argumenty. Opis funkcji +(400) przedstawiono poniżej [3].

+(400) – symbol funkcji	+(400)
Au – pierwszy składnik	Au
Ad – drugi składnik	Ad
R – suma	R

Tabela 1. 2. Specyfikacja argumentów [1]

Obszar	Au	Ad	R
CIO	CIO 0000 do CIO 6143		
Roboczy	W000 do W511		
Podtrzymywania	H000 do H511		
Pomocniczy	A000 do A959		A448 do A959
Timer-ów	T0000 do T4095		
Counter-ów	C0000 do C4095		
Danych	D00000 do D32767		
Powiększony obszar pamięci bez banku	E00000 do E32767		
Powiększony obszar pamięci z bankiem	En_00000 do En_32767 (n=0 do C)		
Stałe	#0000 do #FFFF		

Na rys. 1.1 przedstawiony został program realizujący sumę dwóch liczb. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D100 za pomocą instrukcji BSET. Następnie do rejestru D1 wprowadzona została wartość 2, a do D2 wartość 10. Funkcja sumy została zrealizowana dodając do siebie rejestry D1 i D2, otrzymując wynik w rejestrze D3.

Rys. 1.1. Dodawanie liczb 16b bez przeniesienia

Odejmowanie liczb 16b bez przeniesienia

Funkcja -(410) realizuje odejmowanie liczb szesnastobitowych hexadecymalnych z zakresu #0000 do #FFFF. Funkcja ta zawiera 3 argumenty. Opis funkcji -(410) przedstawiono poniżej [3].

-(410) – symbol funkcji

Mi – odjemna

Su – odjemnik

R – wynik

-(410)
Mi
Su
R

Specyfikacja argumentów jest taka sama jak w tabeli 1.2. przy czym oznaczenie Mi odpowiada Au, a Su odpowiada Ad.

Na rys. 1.2 przedstawiony został program realizujący różnicę dwóch liczb. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D100 za pomocą instrukcji BSET. Następnie do rejestru D1 wprowadzona została wartość 2, a do D2 wartość 10. Funkcja różnicy została zrealizowana odejmując od siebie rejestry D1 i D2 otrzymując wynik w rejestrze D3.

Rys. 1.2. Odejmowanie liczb 16b bez przeniesienia

Mnożenie liczb 16b ze znakiem

Funkcja *(420) realizuje mnożenie liczb szesnastobitowych hexadecymalnych z zakresu #0000 do #FFFF. Funkcja ta zawiera 3 argumenty. Opis funkcji *(420) przedstawiono poniżej [3].

*(420) – symbol funkcji

Md – mnożna

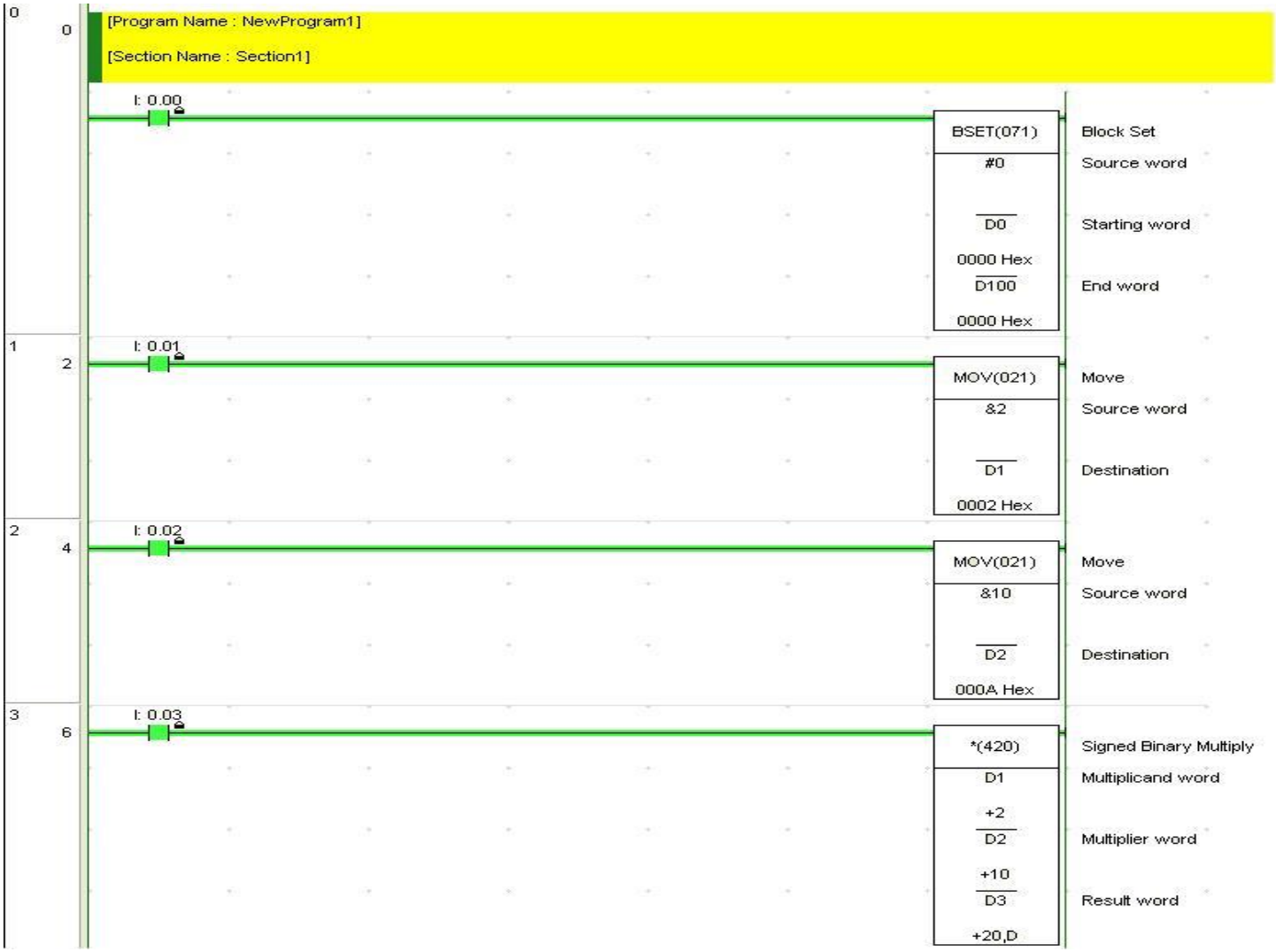
Mr – mnożnik

R – wynik

*(420)
Md
Mr
R

Specyfikacja argumentów jest taka sama jak w tabeli 1.2 przy czym oznaczenie Md odpowiada Au natomiast Mr odpowiada Ad.

Na rys. 1.3 przedstawiony został program realizujący iloczyn dwóch liczb. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D100 za pomocą instrukcji BSET. Następnie do rejestru D1 wprowadzona została wartość 2, a do D2 wartość 10. Funkcja iloczynu została zrealizowana mnożąc przez siebie rejestry D1 i D2 otrzymując wynik w rejestrze D3.



Rys. 1.3. Mnożenie liczb 16b ze znakiem

Dzielenie liczb 16b ze znakiem

Funkcja /(430) realizuje dzielenie liczb szesnastobitowych hexadecymalnych z zakresu #0000 do #FFFF. Funkcja ta zawiera 3 argumenty. Opis funkcji /(430) przedstawiono poniżej [3].

/(430) – symbol funkcji

Dd – dzielna

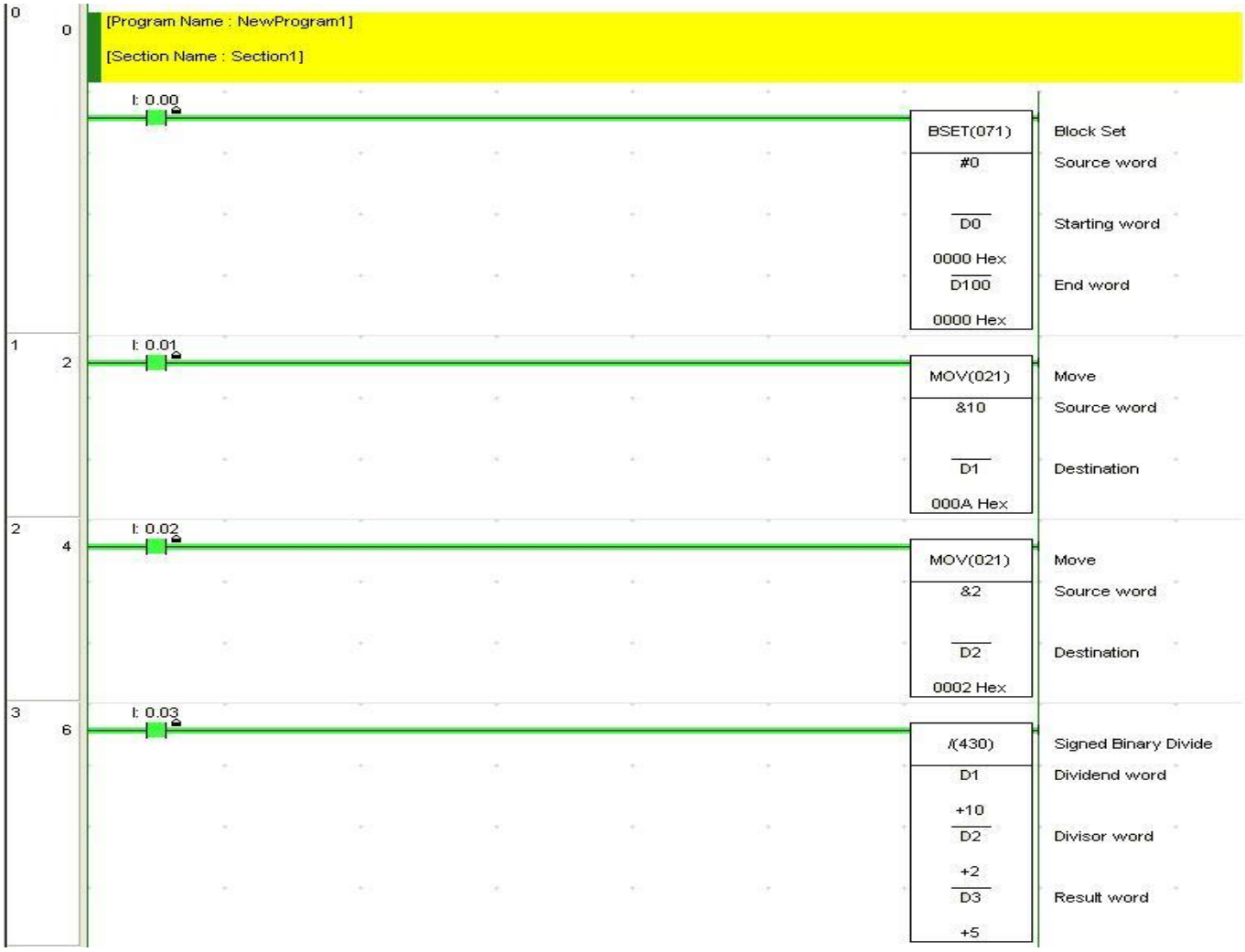
Dr – dzielnik

R – wynik

/(430)
Dd
Dr
R

Specyfikacja argumentów jest taka sama jak w tabeli 1.2 przy czym oznaczenie Dd odpowiada Au natomiast Dr odpowiada Ad.

Na rys. 1.4 przedstawiony został program realizujący iloraz dwóch liczb. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D100 za pomocą instrukcji BSET. Następnie do rejestru D1 wprowadzona została wartość 10, a do D2 wartość 2. Funkcja ilorazu została zrealizowana dzieląc przez siebie rejestry D1 i D2 otrzymując wynik w rejestrze D3.



Rys. 1.4. Dzielenie liczb 16b ze znakiem

Dodawanie liczb 16b BCD z przeniesieniem

Funkcja +BC(406) realizuje dodawanie liczb szesnastobitowych hexadecymalnych z zakresu #0000 do #FFFF. Funkcja ta zawiera 3 argumenty. Opis funkcji +BC(406) przedstawiono poniżej [3].

+BC(406) – symbol funkcji

Au – odjemna

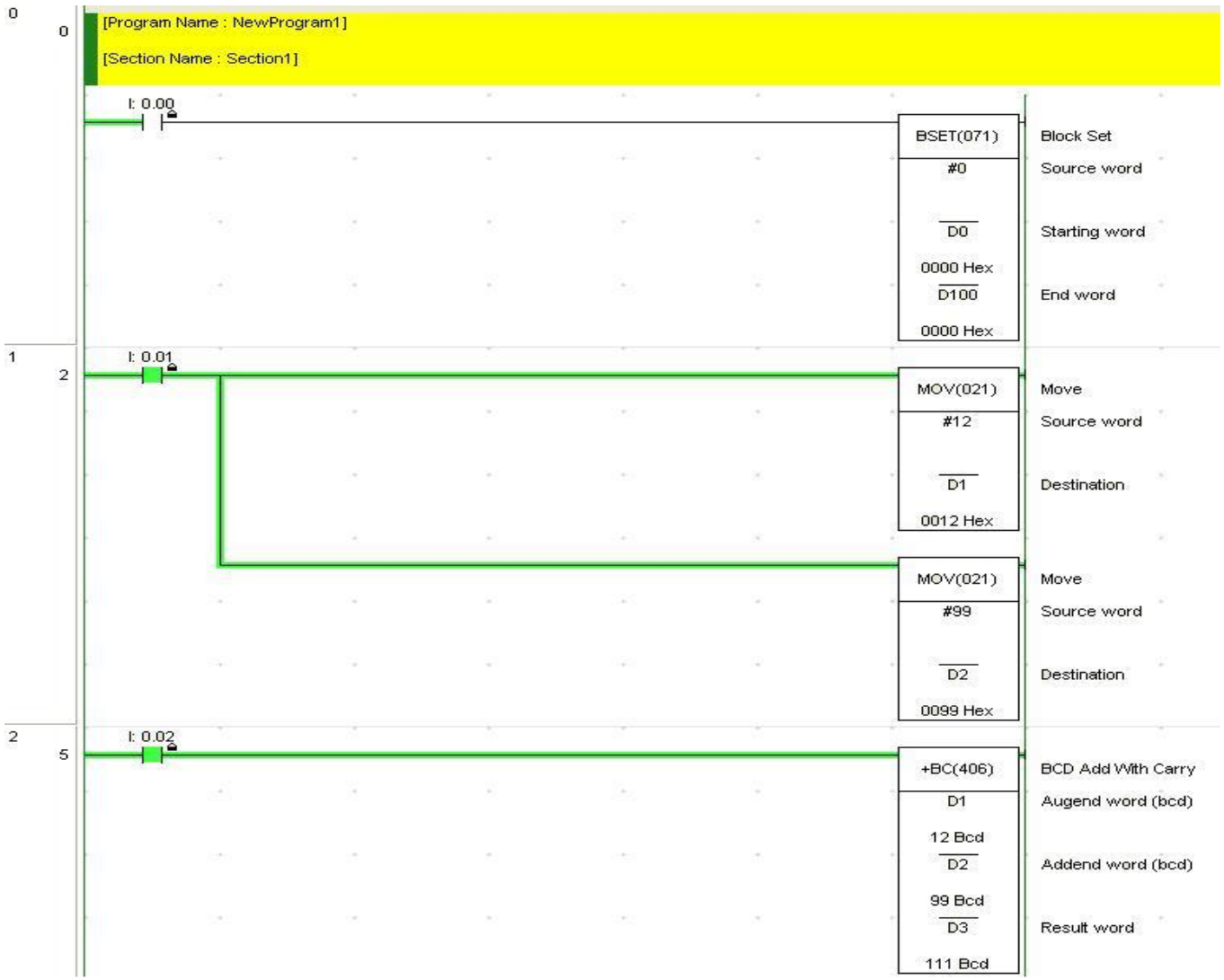
Ad – odjemnik

R – wynik

+BC(406)
Au
Ad
R

Specyfikacja argumentów jest taka sama jak w tabeli 1.2.

Na rys. 1.5 przedstawiony został program realizujący sumę dwóch liczb w kodzie BCD. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D100 za pomocą instrukcji BSET. Następnie do rejestru D1 wprowadzona została wartość 12 (szesnastkowo), a do D2 wartość 99 (szesnastkowo). Funkcja sumy została zrealizowana dodając do siebie rejestry D1 i D2 otrzymując wynik w rejestrze D3.



Rys. 1.5. Dodawanie liczb 16b BCD z przeniesieniem

1.2. Operacje zmiennoprzecinkowe

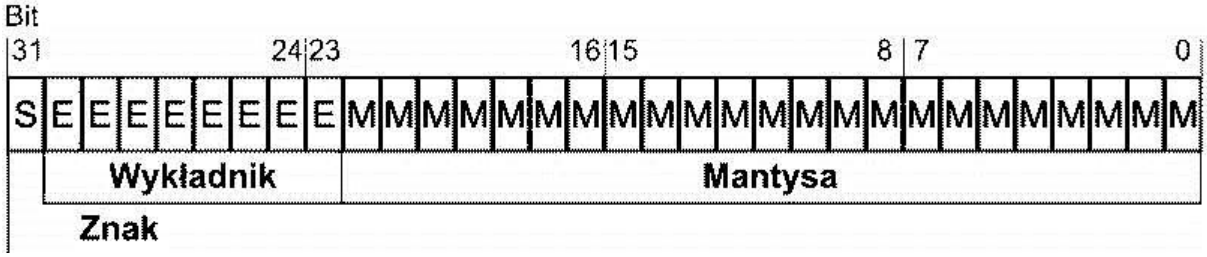
W operacjach zmiennoprzecinkowych wszystkie dane muszą być zamienione na zmiennoprzecinkowe trzydziestodwu- lub szesnastobitowe i zapisane w dwóch rejestrach.

Format zapisywania liczb rzeczywistych jest następujący:

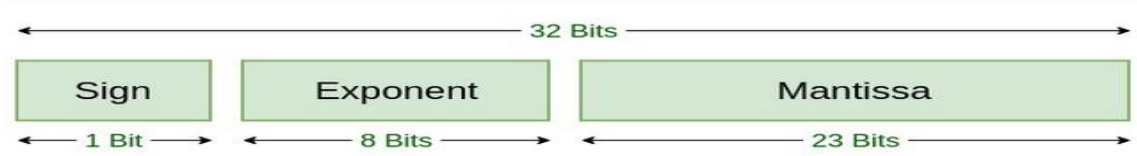
$X = (-1)^s 2^{e-127} (1.f)$

Gdzie: s – znak liczby, e – wykładnik potęgi, f – mantysa.

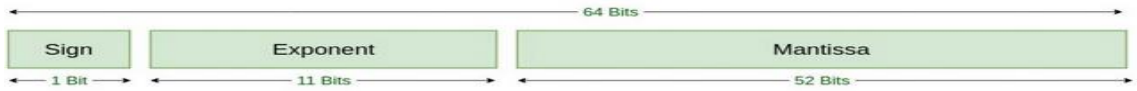
Liczba pojedynczej precyzji (**32b**) zostaje zapisana na trzydziestu dwóch bitach w następujący sposób [3]:



Bity: **0 do 22** to mantysa, **23 do 30** to wykładnik potęgi, **31** to znak. Poniżej przykład



Single Precision
IEEE 754 Floating-Point Standard



Double Precision
IEEE 754 Floating-Point Standard

TYPES	SIGN	BIASED EXPONENT	NORMALISED MANTISA	BIAS

Format zapisu zmiennoprzecinkowego IEEE 754			
32 bity - pojedyncza precyzja	(1 bit) b_{31}	(8 bitów) $b_{30} \dots b_{23}$ (BIAS=127)	(23 bity) ,$b_{22} \dots b_0$ (U1)
64 bity - podwójna precyzja	(1 bit) b_{63}	(11 bitów) $b_{62} \dots b_{52}$ (BIAS=1023)	(52 bity) ,$b_{51} \dots b_0$ (U1)
Opis pół bitowych	bit znaku	bity kodu cechy	bity ułamkowe mantysy

```
85.125
85 = 1010101
0.125 = 001
85.125 = 1010101.001
       =1.010101001 x 2^6
sign = 0

1. Single precision:
biased exponent 127+6=133
133 = 10000101
Normalised mantisa = 010101001
we will add 0's to complete the 23 bits

The IEEE 754 Single precision is:
= 0 10000101 01010100100000000000000
This can be written in hexadecimal form 42AA4000

2. Double precision:
biased exponent 1023+6=1029
1029 = 10000000101
Normalised mantisa = 010101001
we will add 0's to complete the 52 bits

The IEEE 754 Double precision is:
= 0 10000000101 0101010010000000000000000000000000000000000000000000
This can be written in hexadecimal form 4055480000000000
```

Sterowniki PLC umożliwiają realizację operacji matematycznych (zmiennoprzecinkowych) przedstawionych w tabeli 1.3 [1].

Tabela 1.3. Operacje matematyczne zmiennoprzecinkowe [1]

Symbol funkcji	Kod funkcji	Nazwa funkcji
FIX	(450)	Zamiana liczby zmiennoprzecinkowej na 16b.
FIXL	(451)	Zamiana liczby zmiennoprzecinkowej na 32b.
FLT	(452)	Zamiana liczby 16b na zmiennoprzecinkową.
FLTL	(453)	Zamiana liczby 32b na zmiennoprzecinkową.
+F	(454)	Dodawanie liczb zmiennoprzecinkowych
-F	(455)	Odejmowanie liczb zmiennoprzecinkowych
*F	(456)	Mnożenie liczb zmiennoprzecinkowych
/F	(457)	Dzielenie liczb zmiennoprzecinkowych
RAD	(458)	Zamiana stopni na radiany
DEG	(459)	Zamiana radianów na stopnie
SIN	(460)	Obliczanie sinusa kąta
COS	(461)	Obliczanie cosinusa kąta
TAN	(462)	Obliczanie tangensa kąta
ASIN	(463)	Obliczanie arcus sinus kąta
ACOS	(464)	Obliczanie arcus cosinus kąta
ATAN	(465)	Obliczanie arcus tangens kąta
SQRT	(466)	Obliczanie pierwiastka kwadratowego
EXP	(467)	Funkcja wykładnicza
LOG	(468)	Obliczanie logarytmu
PWR	(469)	Funkcja eksponencjalna

Poniżej zostaną omówione najczęściej wykorzystywane w obliczeniach funkcje matematyczne.

Zamiana liczby 16b na zmiennoprzecinkową

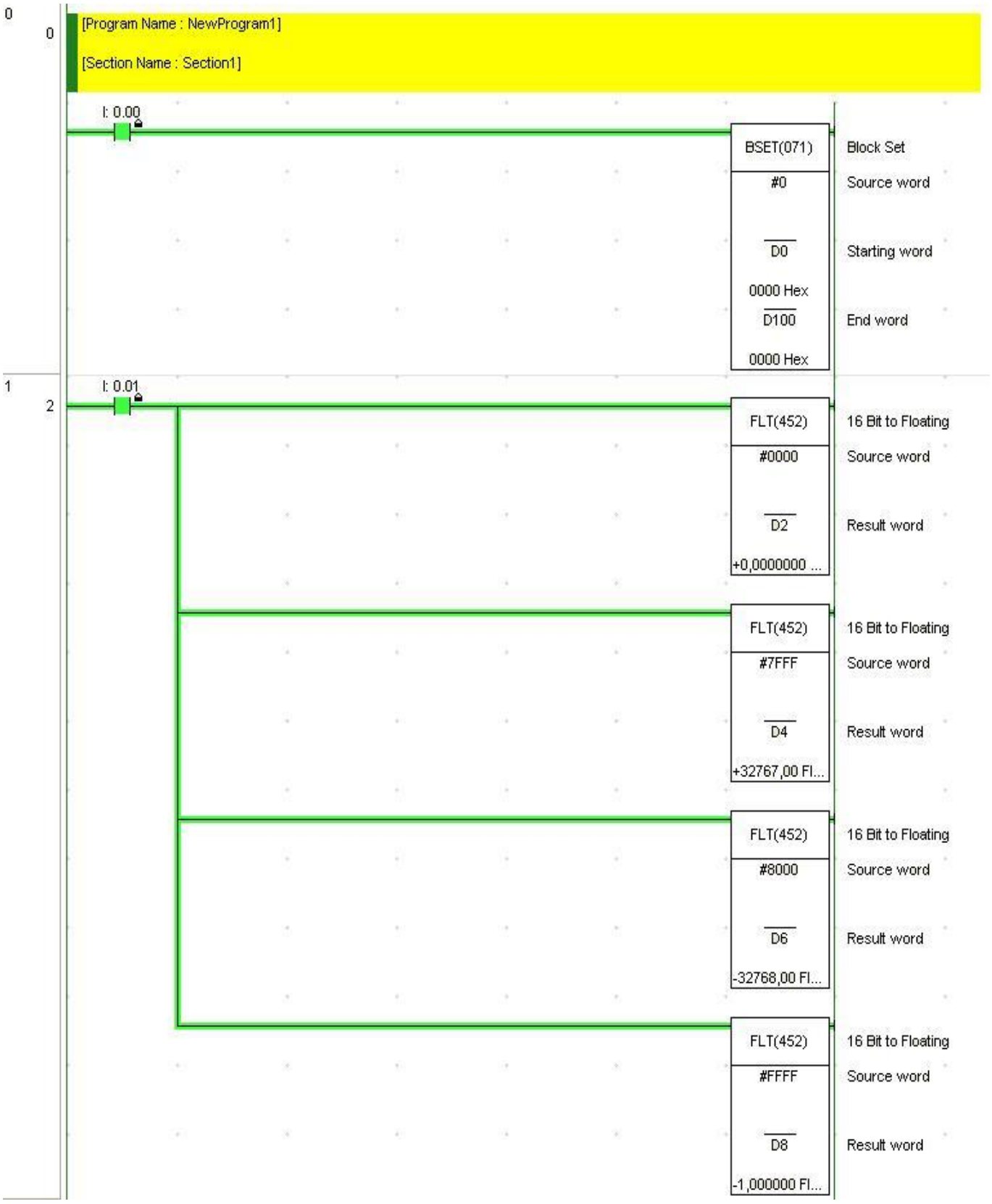
Funkcja FLT(452) realizuje zamianę liczb szesnastobitowych hexadecymalnych z zakresu #0000 do #FFFF na zmiennoprzecinkową. Opis funkcji FLT(452) przedstawiono poniżej [3].

FLT(452) – symbol funkcji
S – wartość źródłowa
R – wynik

FLT(452)
S
R

Funkcja FLT(452) realizuje zamianę danej wartości na szesnastobitową. Dane wprowadza się w kodzie hexadecymalnym (od #0000 do #FFFF), a zakres liczb wynosi od –32768 do 32767. Jeśli po przekonwertowaniu należy otrzymać liczbę z zakresu 0 – 32767 (dodatnie), to na wejście funkcji należy podać wartości #0000 - #7FFF (gdzie #7FFF odpowiada 32767 dziesiętnie). W celu otrzymania wartości ujemnych należy wpisywać liczby od #8000 do #FFFF.

Na rys. 1.6 przedstawiony został program realizujący zamianę liczb szesnastobitowych na zmiennoprzecinkowe. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D100 za pomocą instrukcji BSET. Następnie pokazano zastosowanie funkcji FLT(452) i przekonwertowane wartości zapisano w kolejnych rejestrach D2, D4, D6, D8 pamiętając, że każda liczba zajmuje 2 rejestry.



Rys. 1.6. Zamiana liczby 16b na zmiennoprzecinkową

Dodawanie liczb zmiennoprzecinkowych

Funkcja +F(454) realizuje dodawanie liczb trzydziestodwubitowych hexadecymalnych z zakresu #00000000 do #FFFFFFFF. Funkcja ta zawiera 3 argumenty. Opis funkcji +F(454) przedstawiono poniżej [3].

+F(454) – symbol funkcji
Au – pierwszy składnik
Ad – drugi składnik
R – suma

+F(454)
Au
Ad
R

Tabela 1.4. Specyfikacja argumentów [2]

Obszar	Au	Ad	R
CIO	CIO 0000 do CIO 6142		
Roboczy	W000 do W510		
Przechowywania bitu	H000 do H510		
Pomocniczego bitu	A000 do A958		A448 do A958
Timer	T0000 do T4094		
Counter	C0000 do C4094		
Przepływu danych	D00000 do D32766		
Powiększony obszar pamięci bez banku	E00000 do E32766		
Powiększony obszar pamięci z bankiem	En_00000 do En_32766 (n=0 do C)		
Stałe	#00000000 do #FFFFFFFF		

Na rys. 1.7 przedstawiony został program realizujący sumę dwóch liczb zmiennoprzecinkowych. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D100 za pomocą instrukcji BSET. Następnie do rejestru D2 wprowadzona została wartość 2. Za pomocą funkcji FLT(452) liczba ta została zamieniona na zmiennoprzecinkową i zapisana w rejestrze D20. W kolejnym kroku do rejestru D6 wprowadzono liczbę 10. Analogicznie jak w poprzednim przypadku za pomocą funkcji FLT(452) zamieniono ją na zmiennoprzecinkową i zapisano w D60. Funkcja sumy została zrealizowana dodając do siebie rejestry D20 i D60 otrzymując wynik w rejestrze D80.

Rys 1.7 Dodawanie liczb zmiennoprzecinkowych

Odejmowanie liczb zmiennoprzecinkowych

Funkcja -F(455) realizuje odejmowanie liczb trzydziestodwubitowych hexadecymalnych z zakresu #00000000 do #FFFFFFFF. Funkcja ta zawiera 3 argumenty. Opis funkcji -F(455) przedstawiono poniżej [3].

-F(455) – symbol funkcji
Mi – odjemna
Su – odjemnik
R – wynik

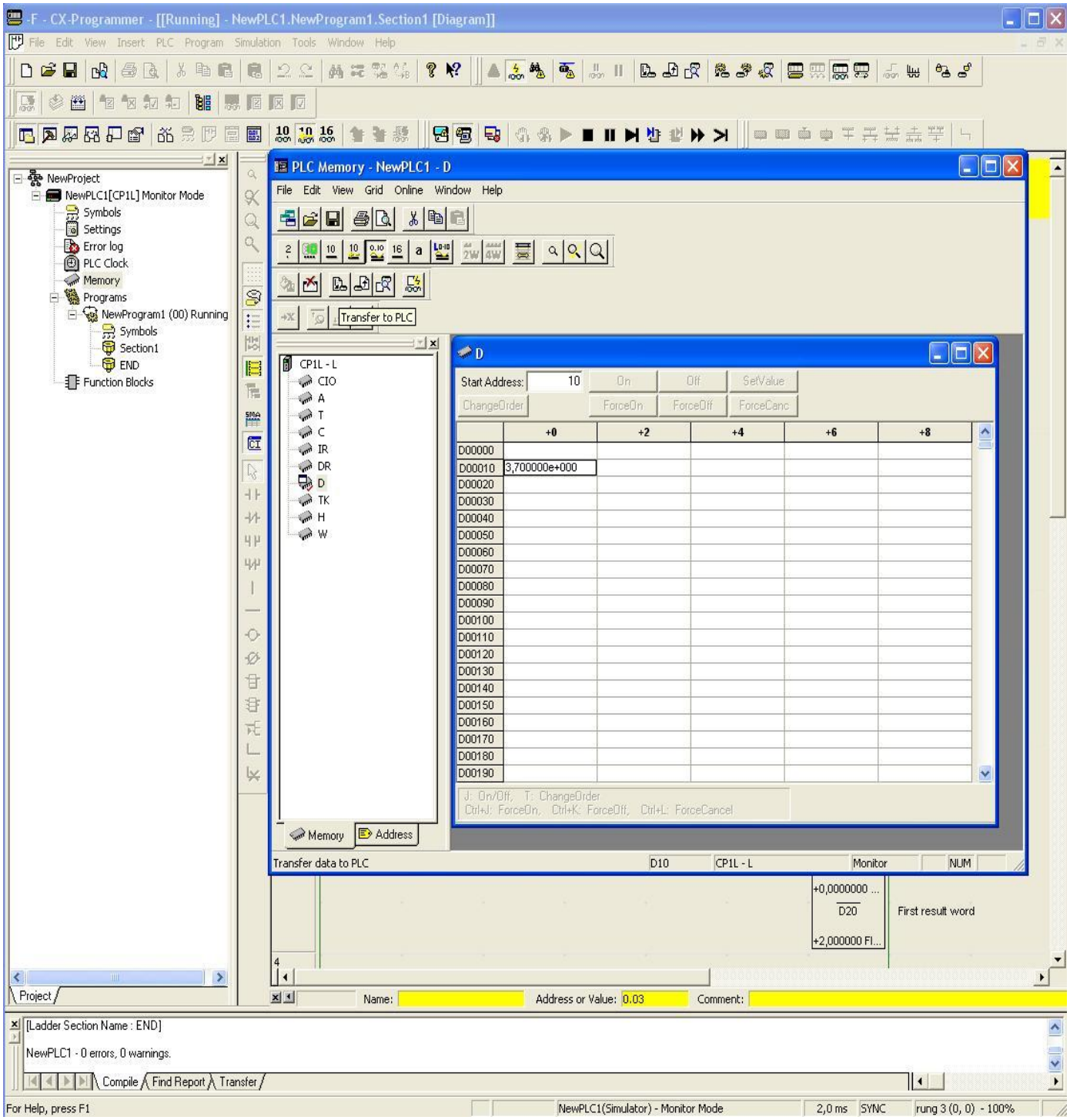
-F(455)
Mi
Su
R

Specyfikacja argumentów jest taka sama jak w tabeli 1.4. przy czym oznaczenie Mi odpowiada Au natomiast Su odpowiada Ad.

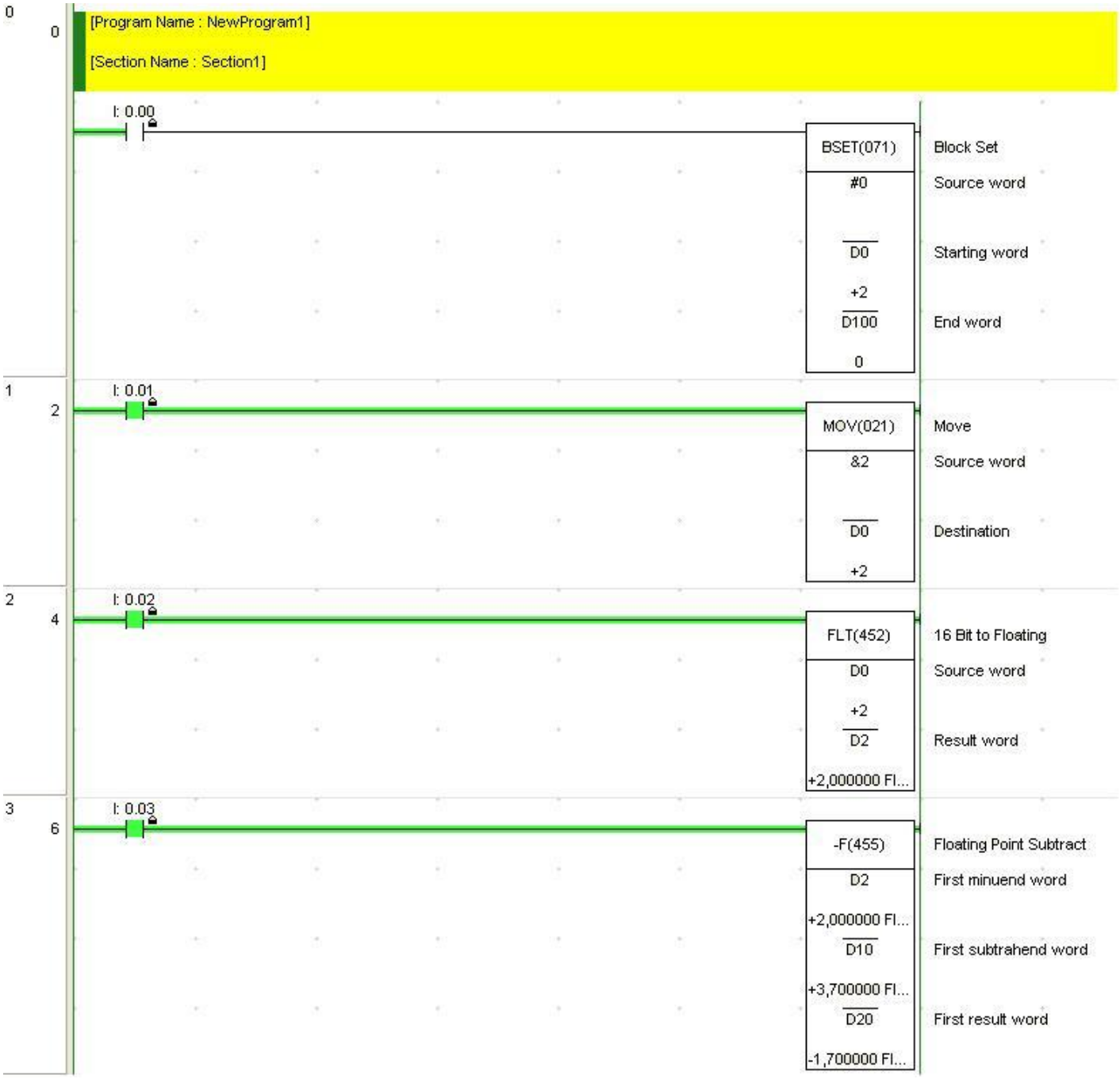
Na rys. 1.8 przedstawiony został program realizujący różnicę dwóch liczb zmiennoprzecinkowych. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D100 za pomocą instrukcji BSET. Następnie do rejestru D0 wprowadzona została wartość 2. Z zastosowaniem funkcji FLT(452) liczba ta została zamieniona na zmiennoprzecinkową i zapisana w rejestrze D2.

Następna liczba została wprowadzona bezpośrednio do komórki pamięci D10 tak jak pokazano to na rysunku 1.8. Można to wykonać po przejściu w zakładkę *Memory* i otwarciu rejestru D. W wyznaczoną komórkę należy wpisać dowolną liczbę. Po wpisaniu zadanej wartości należy przesłać dane do sterownika PLC. W tym celu wybieramy przycisk „transfer to PLC”.

Funkcja różnicy została zrealizowana odejmując od siebie rejestry D2 i D10 otrzymując wynik w D20.



Rys. 1.8. Wprowadzanie liczby bezpośrednio do komórki



Rys. 1.9. Odejmowanie liczb zmiennoprzecinkowych

Mnożenie liczb zmiennoprzecinkowych

Funkcja *F(456) realizuje mnożenie liczb trzydziestodwubitowych hexadecymalnych z zakresu #00000000 do #FFFFFFFF. Funkcja ta zawiera 3 argumenty. Opis funkcji *F(456) przedstawiono poniżej [3].

*F(456) – symbol funkcji

Md – mnożna

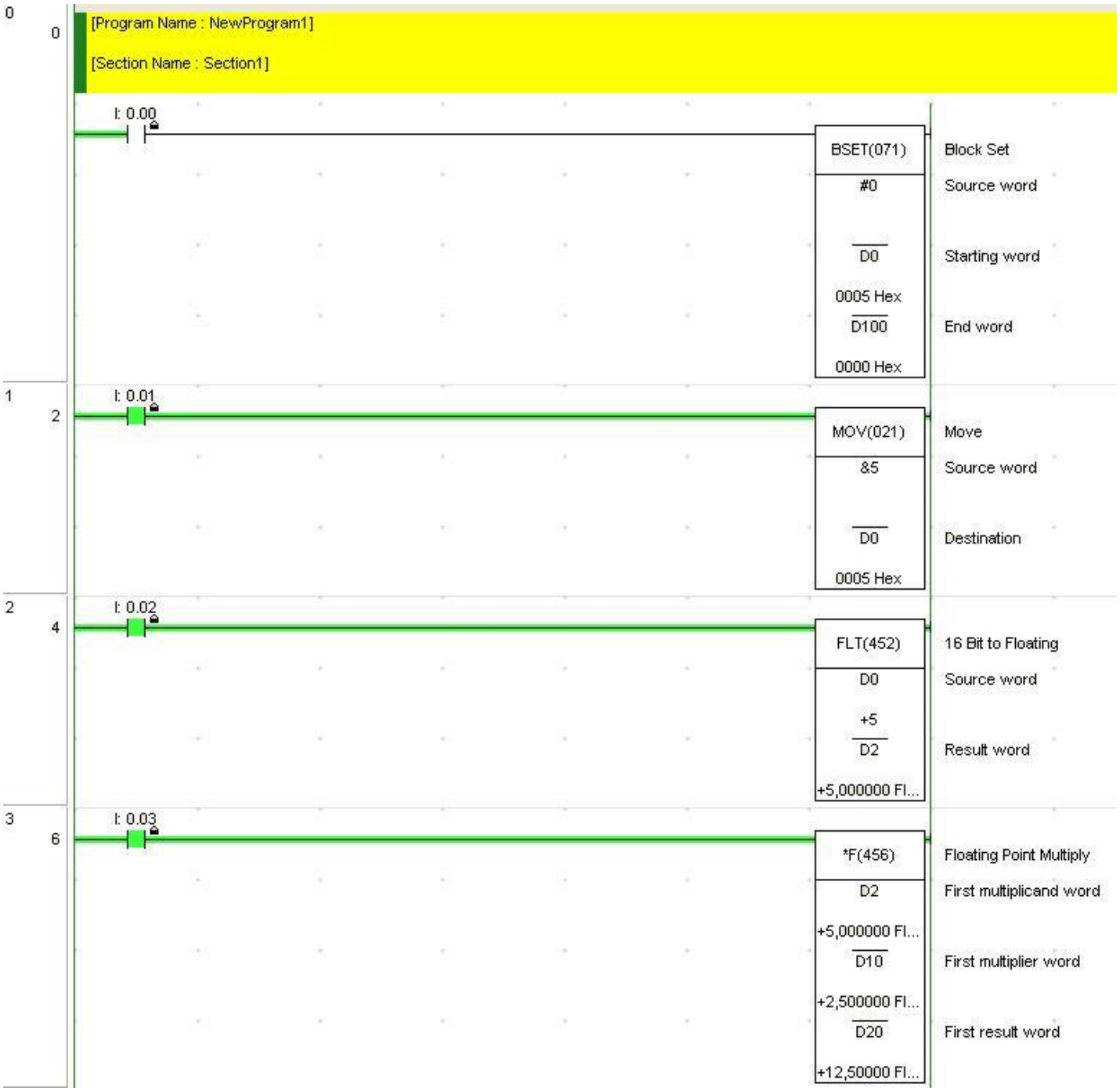
Mr – mnożnik

R – wynik

*F(456)
Md
Mr
R

Specyfikacja argumentów jest taka sama jak w tabeli 1.4. przy czym oznaczenie Md odpowiada Au, natomiast Mr odpowiada Ad.

Na rys. 1.10 przedstawiony został program realizujący iloczyn dwóch liczb zmiennoprzecinkowych. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D100 za pomocą instrukcji BSET. Następnie do rejestru D0 wprowadzona została wartość 5. Z zastosowaniem funkcji FLT(452) liczba ta została zamieniona na zmiennoprzecinkową i zapisana w rejestrze D2. Druga wartość została wprowadzona bezpośrednio do komórki pamięci, tak jak to przedstawia rys 1.10 przy czym do rejestru D10 wprowadzono liczbę 2,5. Funkcja iloczynu została zrealizowana mnożąc przez siebie rejestry D2 i D10 otrzymując wynik w D20.



Rys. 1.10. Mnożenie liczb zmiennoprzecinkowych

Dzielenie liczb zmiennoprzecinkowych

Funkcja /F(457) realizuje dzielenie liczb trzydziestodwubitowych hexadecymalnych z zakresu #00000000 do #FFFFFFFF. Funkcja ta zawiera 3 argumenty. Opis funkcji /F(457) przedstawiono poniżej.

/F(457) – symbol funkcji	/F(457)
Dd – dzielna	Dd
Dr – dzielnik	Dr
R – wynik	R

Specyfikacja argumentów jest taka sama jak w tabeli 1.4 przy czym oznaczenie Dd odpowiada Au natomiast Dr odpowiada Ad.

Na rys. 1.11 przedstawiony został program realizujący iloraz dwóch liczb zmiennoprzecinkowych. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D100 za pomocą instrukcji BSET. Następnie do rejestru D0 wprowadzona została wartość 5. Z zastosowaniem funkcji FLT(452) liczba ta została zamieniona na zmiennoprzecinkową i zapisana w rejestrze D2. W kolejnym kroku wpisano wartość 15 do rejestru D4. Analogicznie została ona zamieniona na zmiennoprzecinkową i zapisana w rejestrze D6. Funkcja iloczynu została zrealizowana dzieląc przez siebie rejestry D2 i D6 otrzymując wynik w D20.

Rys. 1.11. Dzielenie liczb zmiennoprzecinkowych

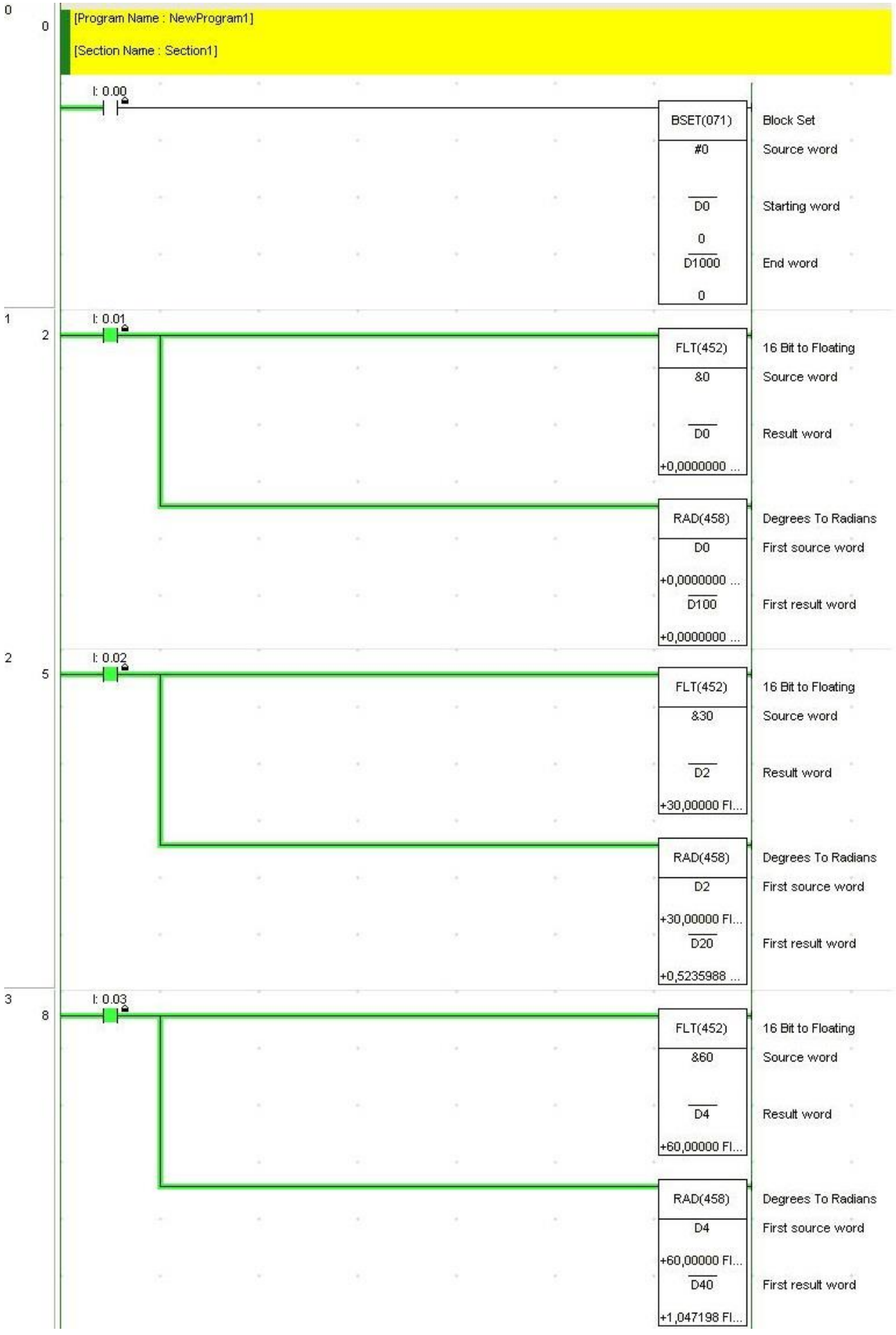
Zamiana stopni na radiany

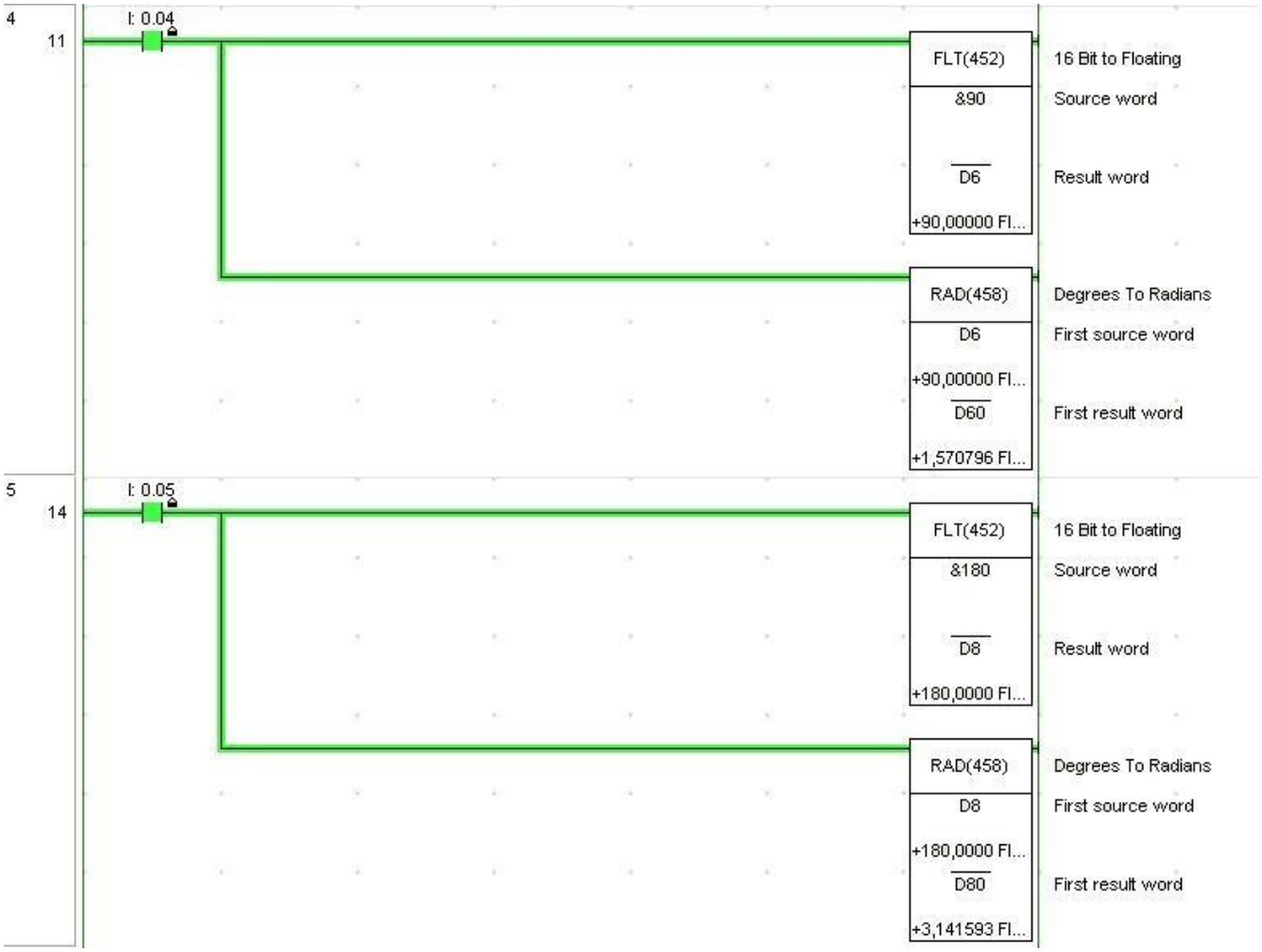
Funkcja RAD(458) realizuje konwersję liczb trzydziestodwubitowych hexadecymalnych z zakresu #00000000 do #FFFFFFFF ze stopni na radiany. Funkcja ta zawiera 2 argumenty. Opis funkcji RAD(458) przedstawiono poniżej [3].

RAD(458) – symbol funkcji	RAD(458)
S – wartość kąta w stopniach	S
R – wynik	R

Specyfikacja argumentów jest taka sama jak w tabeli 1.4. przy czym oznaczenie S odpowiada Au.

Na rys. 1.12 przedstawiony został program realizujący zamianę stopni na radiany. Instrukcja wykonywana jest na liczbach zmiennoprzecinkowych. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D1000 za pomocą instrukcji BSET. Przykład wykorzystania instrukcji został zrealizowany dla najczęściej używanych kątów takich jak: 0°, 30°, 60°, 90°, 180°. Wszystkie wartości na wejściu funkcji RAD(458) muszą mieć wartość zmiennoprzecinkową. W celu poprawnego działania programu wszystkie podane wyżej wartości kątów zostały zamienione na zmiennoprzecinkowe za pomocą funkcji FLT(452).


0
0
[Program Name : NewProgram1]
[Section Name : Section1]
I: 0.00
BSET(071)
Block Set
#0
Source word
D0
Starting word
0
D1000
End word
0
1
2
I: 0.01
FLT(452)
16 Bit to Floating
&0
Source word
D0
Result word
+0,0000000 ...
RAD(458)
Degrees To Radians
D0
First source word
+0,0000000 ...
D100
First result word
+0,0000000 ...
2
5
I: 0.02
FLT(452)
16 Bit to Floating
&30
Source word
D2
Result word
+30,00000 Fl...
RAD(458)
Degrees To Radians
D2
First source word
+30,00000 Fl...
D20
First result word
+0,5235988 ...
3
8
I: 0.03
FLT(452)
16 Bit to Floating
&60
Source word
D4
Result word
+60,00000 Fl...
RAD(458)
Degrees To Radians
D4
First source word
+60,00000 Fl...
D40
First result word
+1,047198 Fl...



Rys. 1.12. Zamiana stopni na radiany

Obliczanie sinusa kąta

Funkcja SIN(460) oblicza sinus liczby trzydziestodwubitowej hexadecymalnej z zakresu #00000000 do #FFFFFFFF jako wartość podanego kąta w radianach. Funkcja ta zawiera 2 argumenty. Opis funkcji SIN(460) przedstawiono poniżej [3].

SIN(460) – symbol funkcji
S – wartość kąta w stopniach
R – wynik

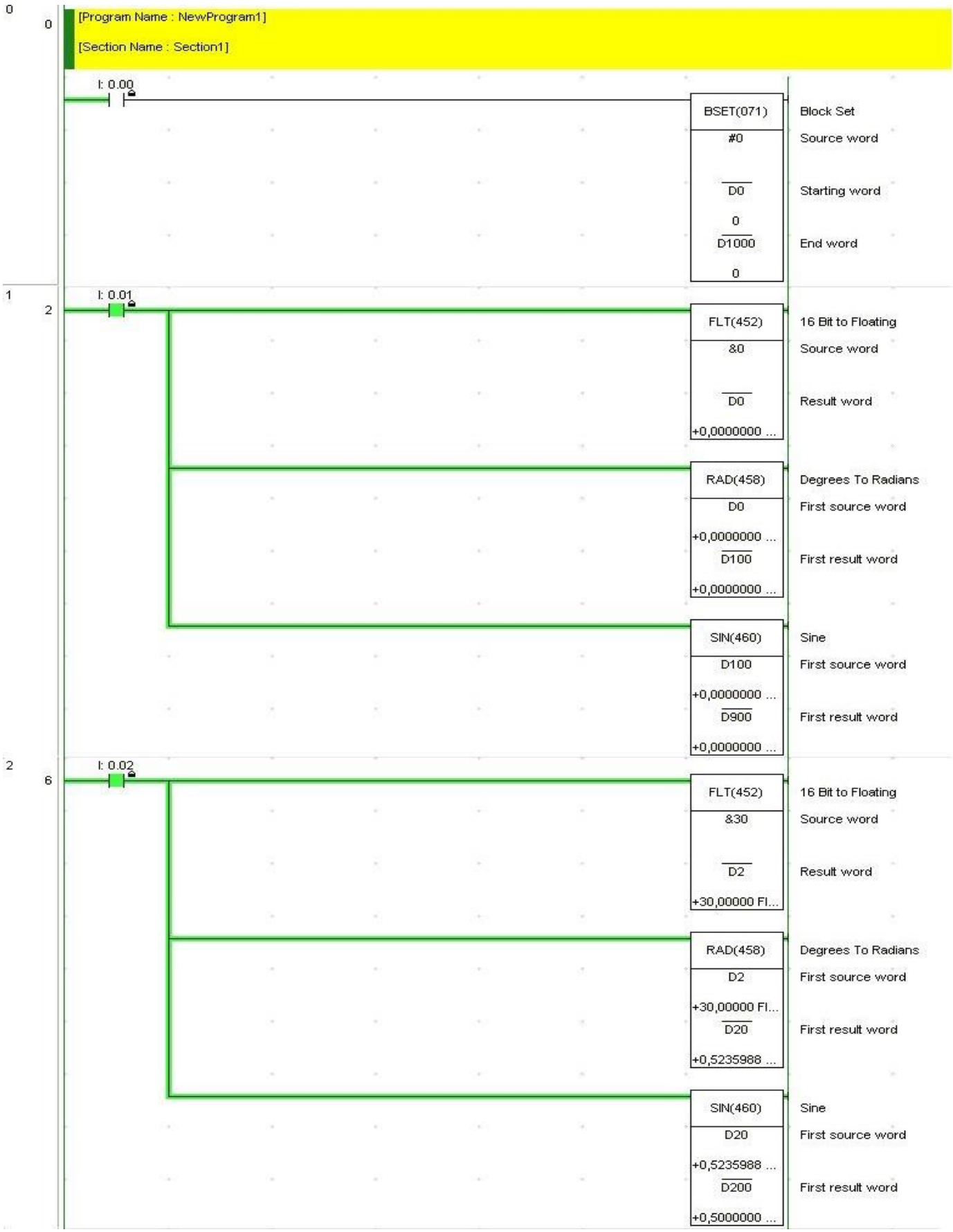
SIN(460)
S
R

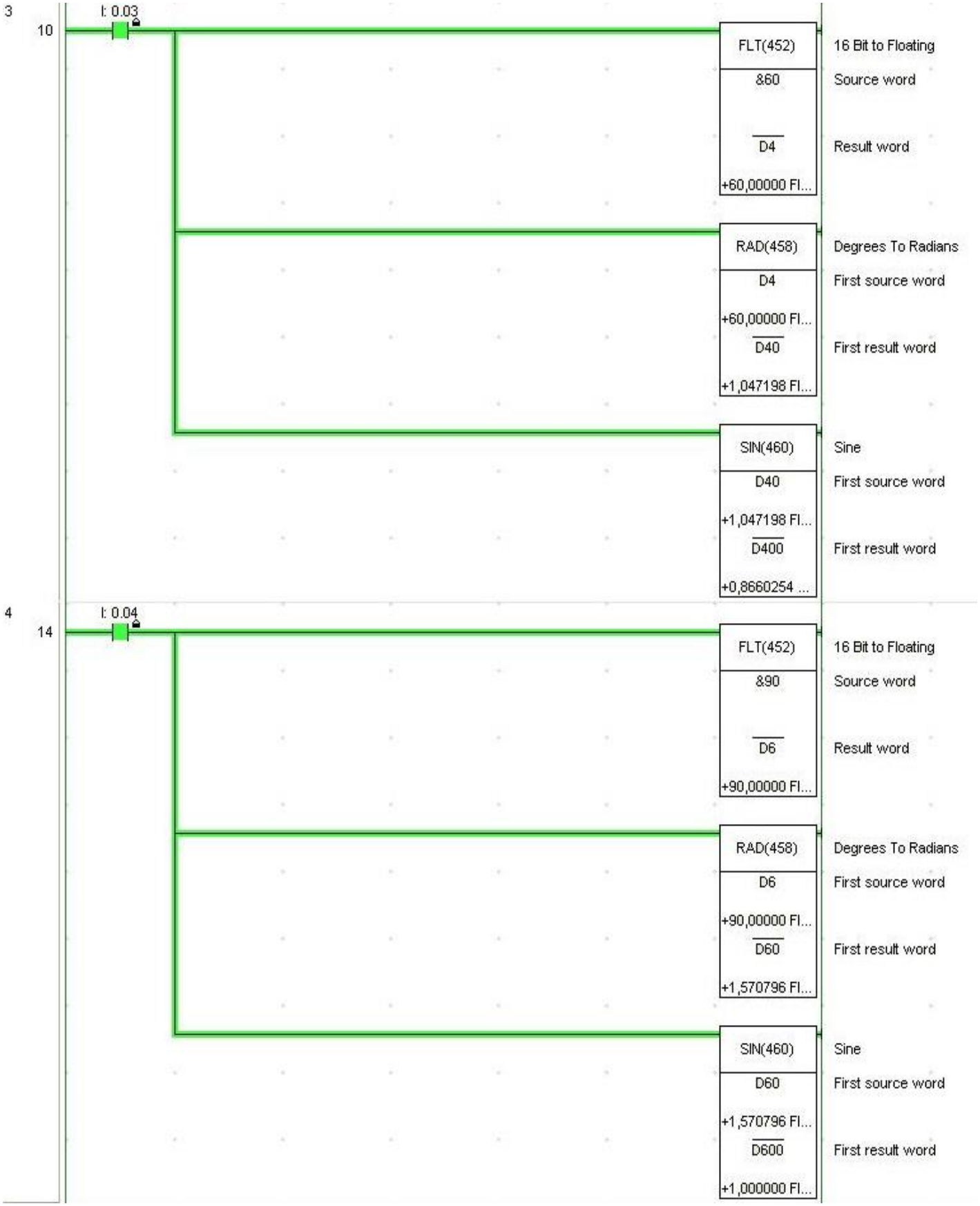
Specyfikacja argumentów jest taka sama jak w tabeli 1.4. przy czym oznaczenie S odpowiada Au.

Na rys. 1.13 przedstawiony został program obliczający sinus kąta. Instrukcja wykonywana jest na liczbach zmiennoprzecinkowych. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D1000 za pomocą instrukcji BSET.

Przykład wykorzystania instrukcji został zrealizowany dla najczęściej używanych kątów takich jak: 0°, 30°, 60°, 90°. Wszystkie wartości na wejściu funkcji SIN(460) muszą mieć wartość zmiennoprzecinkową. W celu poprawnego działania programu wszystkie podane wyżej wartości kątów zostały zamienione na

zmiennoprzecinkowe za pomocą funkcji FLT(452) a następnie przeliczone na radiany. W ostatnim kroku obliczone zostały wartości sinusa dla podanych kątów.





Rys. 1.13. Obliczanie sinusa kąta

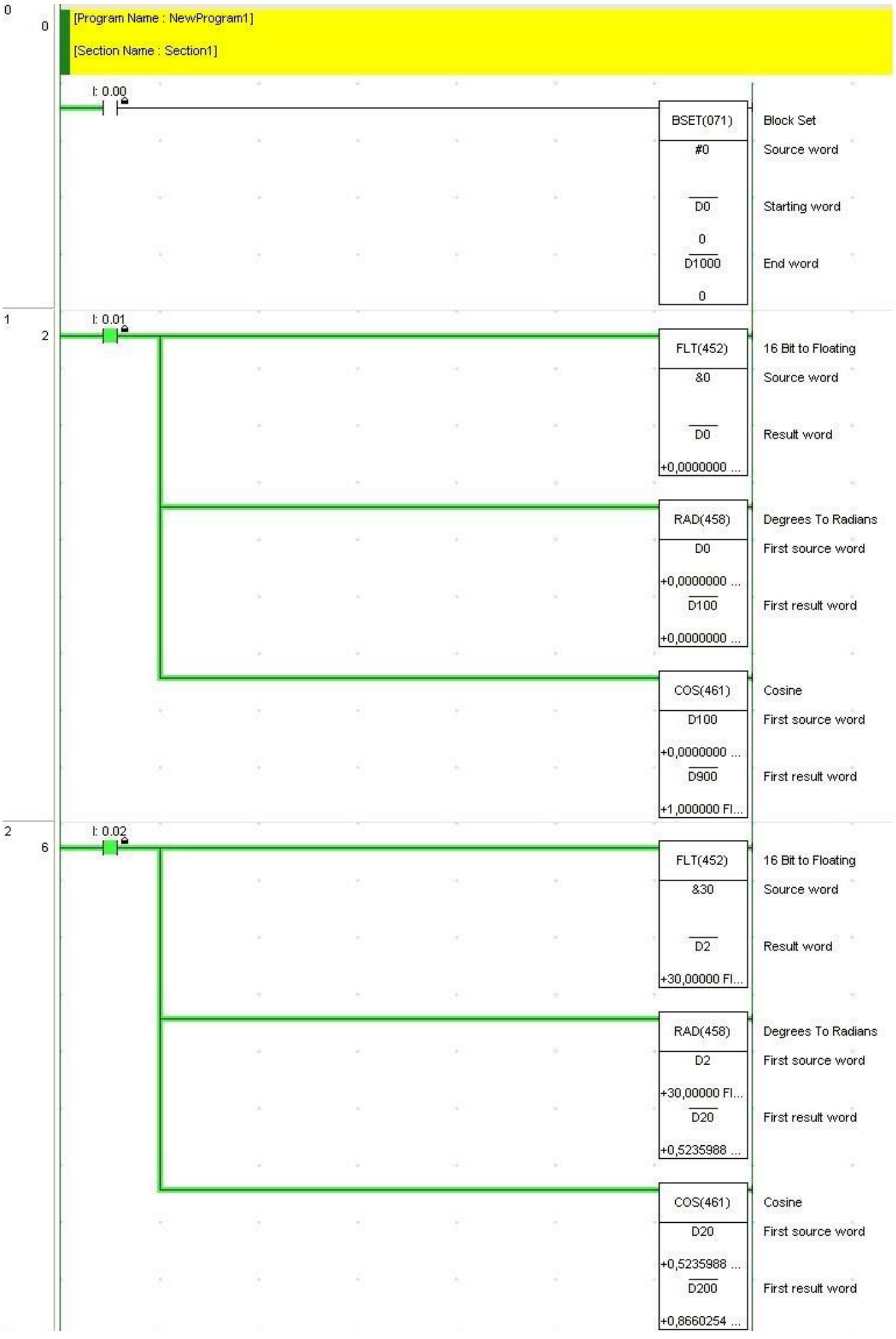
Obliczanie cosinusa kąta

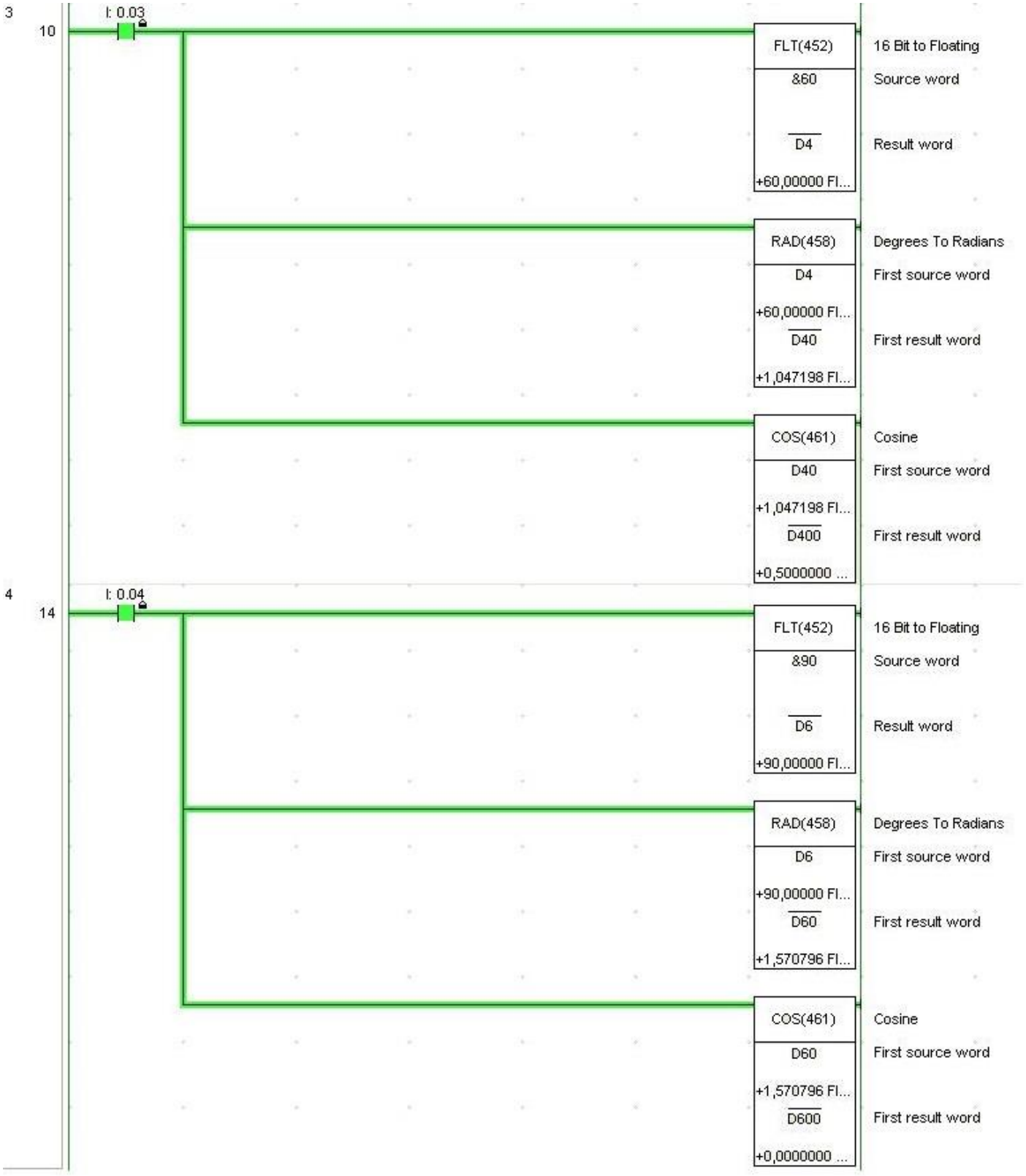
Funkcja COS(461) oblicza cosinus liczby trzydziestodwubitowej hexadecymalnej z zakresu #00000000 do #FFFFFFFF jako wartość podanego kąta w radianach. Funkcja ta zawiera 2 argumenty. Opis funkcji COS(461) przedstawiono poniżej [3].

COS(461) – symbol funkcji	COS(461)
S – wartość kąta w stopniach	S
R – wynik	R

Specyfikacja argumentów jest taka sama jak w tabeli 1.4 przy czym oznaczenie S odpowiada Au.

Na rys. 1.14 przedstawiony został program obliczający cosinus kąta. Instrukcja wykonywana jest na liczbach zmiennoprzecinkowych. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D1000 za pomocą instrukcji BSET. Przykład wykorzystania instrukcji został zrealizowany dla najczęściej używanych kątów takich jak: 0°, 30°, 60° i 90°. Wszystkie wartości na wejściu funkcji COS(461) muszą mieć wartość zmiennoprzecinkową. W celu poprawnego działania programu wszystkie podane wyżej wartości kątów zostały zamienione na zmiennoprzecinkowe za pomocą funkcji FLT(452), a następnie przeliczone na radiany. W ostatnim kroku obliczone zostały wartości cosinusa dla podanych kątów.


0
0
[Program Name : NewProgram1]
[Section Name : Section1]
I: 0.00
BSET(071)
Block Set
#0
Source word
D0
Starting word
0
D1000
End word
0
1
2
I: 0.01
FLT(452)
16 Bit to Floating
&0
Source word
D0
Result word
+0,0000000 ...
RAD(458)
Degrees To Radians
D0
First source word
+0,0000000 ...
D100
First result word
+0,0000000 ...
COS(461)
Cosine
D100
First source word
+0,0000000 ...
D900
First result word
+1,000000 FI...
2
6
I: 0.02
FLT(452)
16 Bit to Floating
&30
Source word
D2
Result word
+30,00000 FI...
RAD(458)
Degrees To Radians
D2
First source word
+30,00000 FI...
D20
First result word
+0,5235988 ...
COS(461)
Cosine
D20
First source word
+0,5235988 ...
D200
First result word
+0,8660254 ...



Rys. 1.14. Obliczanie cosinusa kąta

Część praktyczna pracy

Celem niniejszego projektu jest opracowanie programów w języku drabinkowym na sterownik PLC umożliwiających realizacje wybranych funkcji matematycznych. Aby wykonać tak postawiony w projekcie cel należy wykonać następujące zadania:

- zapoznać się z podstawowymi i zaawansowanymi instrukcjami matematycznymi stałoprzecinkowymi i zmiennoprzecinkowymi realizowanymi przez sterownik PLC,
- zapoznać się ze specyfikacją programowego użycia argumentów występujących w funkcjach matematycznych realizowanymi przez sterownik PLC,
- zapoznać się ze sposobem wprowadzania argumentów do funkcji,
- zapoznać się z programem „CX-Programer", wchodzącym w skład środowiska „Cx-One",
- opracować programy w języku drabinkowym na sterownik PLC umożliwiające realizacje wybranych funkcji matematycznych.

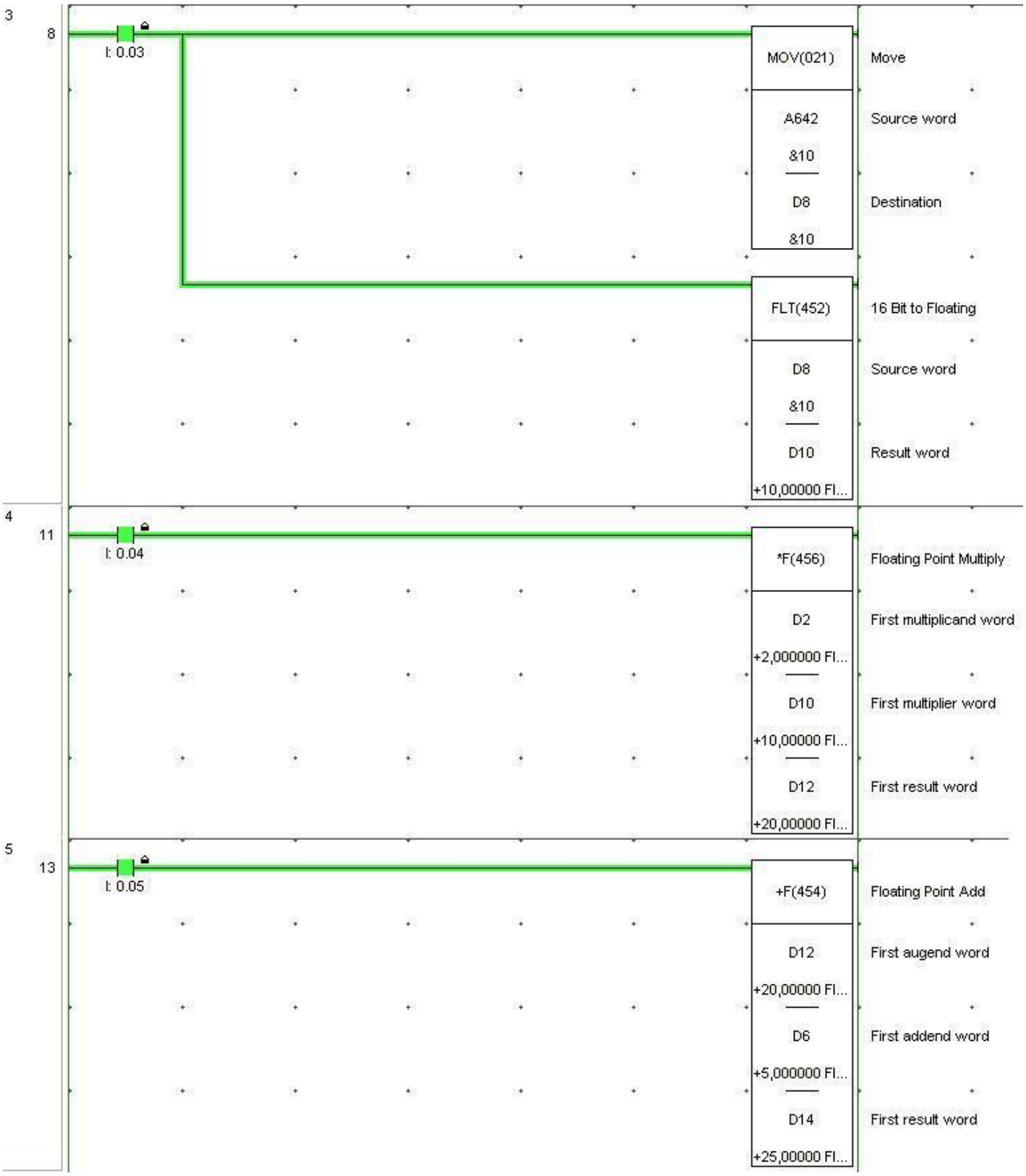
2. Realizacja funkcji matematycznych za pomocą sterownika PLC

W części praktycznej opracowane zostały programy w języku drabinkowym umożliwiające obliczanie wartości wybranych funkcji matematycznych w oparciu o opracowania teoretyczne opisane w rozdziale pierwszym.

2.1. Funkcja liniowa postaci y=ax+b

Na rys. 2.1 przedstawiony został program obliczający wartość zadanej funkcji liniowej. Współczynniki funkcji zostały przyjęte następująco: a=2 oraz b=5. Zmienna x zostaje wprowadzona poprzez komórkę A642. Jest to komórka, której wartość zadaje się pokrętłem analogowym w sterowniku PLC. Jej wartości mogą się zmieniać w przedziale 0-255 ze skokiem co 1. W projekcie ustawiono A642=10. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D100 za pomocą instrukcji BSET. W kolejnym kroku wprowadzona została wartość współczynnika a=2 za pomocą funkcji MOV(021) do rejestru D0. Następnie za pomocą instrukcji FLT(452) liczba ta została zamieniona na zmiennoprzecinkową i zapisana w D2. Analogicznie wpisano współczynnik b=5 do rejestru D6. Wartość zmiennej x zostaje zadana z pokrętła analogowego sterownika i zapisana w D8. Kolejnym krokiem jest zamiana na liczbę zmiennoprzecinkową i zapamiętanie w D10. Następnie wykonane zostało mnożenie zmiennoprzecinkowe z zapisem wyniku w D12. W ostatnim kroku zrealizowano sumowanie wartości rejestrów D12 i D6.


0
0
[Program Name : NewProgram1]
[Section Name : Section1]
I: 0.00
BSET(071)
Block Set
#0
Source word
D0
Starting word
&2
D100
End word
&0
1
2
I: 0.01
MOV(021)
Move
&2
Source word
D0
Destination
&2
FLT(452)
16 Bit to Floating
D0
Source word
&2
D2
Result word
+2,000000 Fl...
2
5
I: 0.02
MOV(021)
Move
&5
Source word
D4
Destination
&5
FLT(452)
16 Bit to Floating
D4
Source word
&5
D6
Result word
+5,000000 Fl...

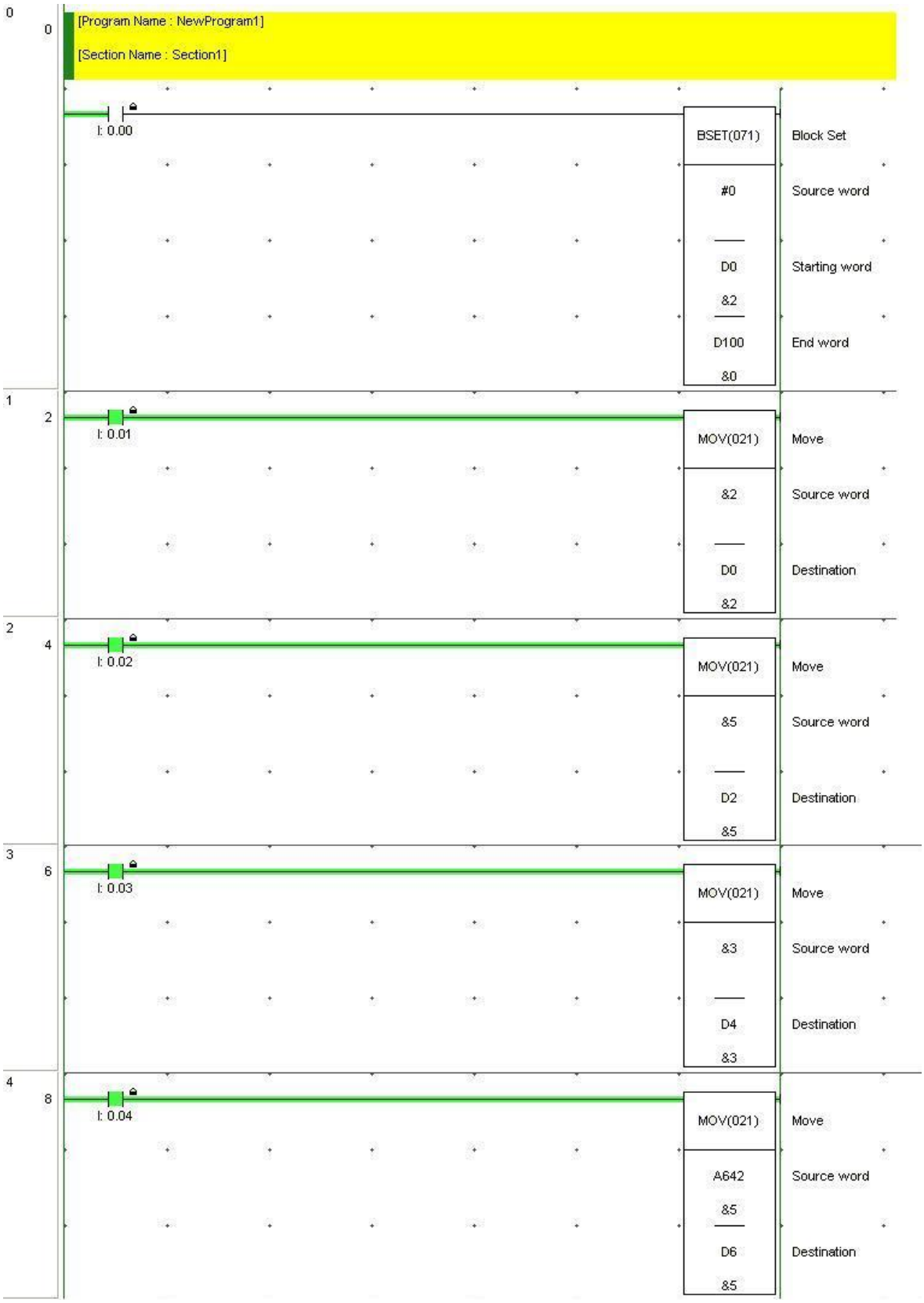


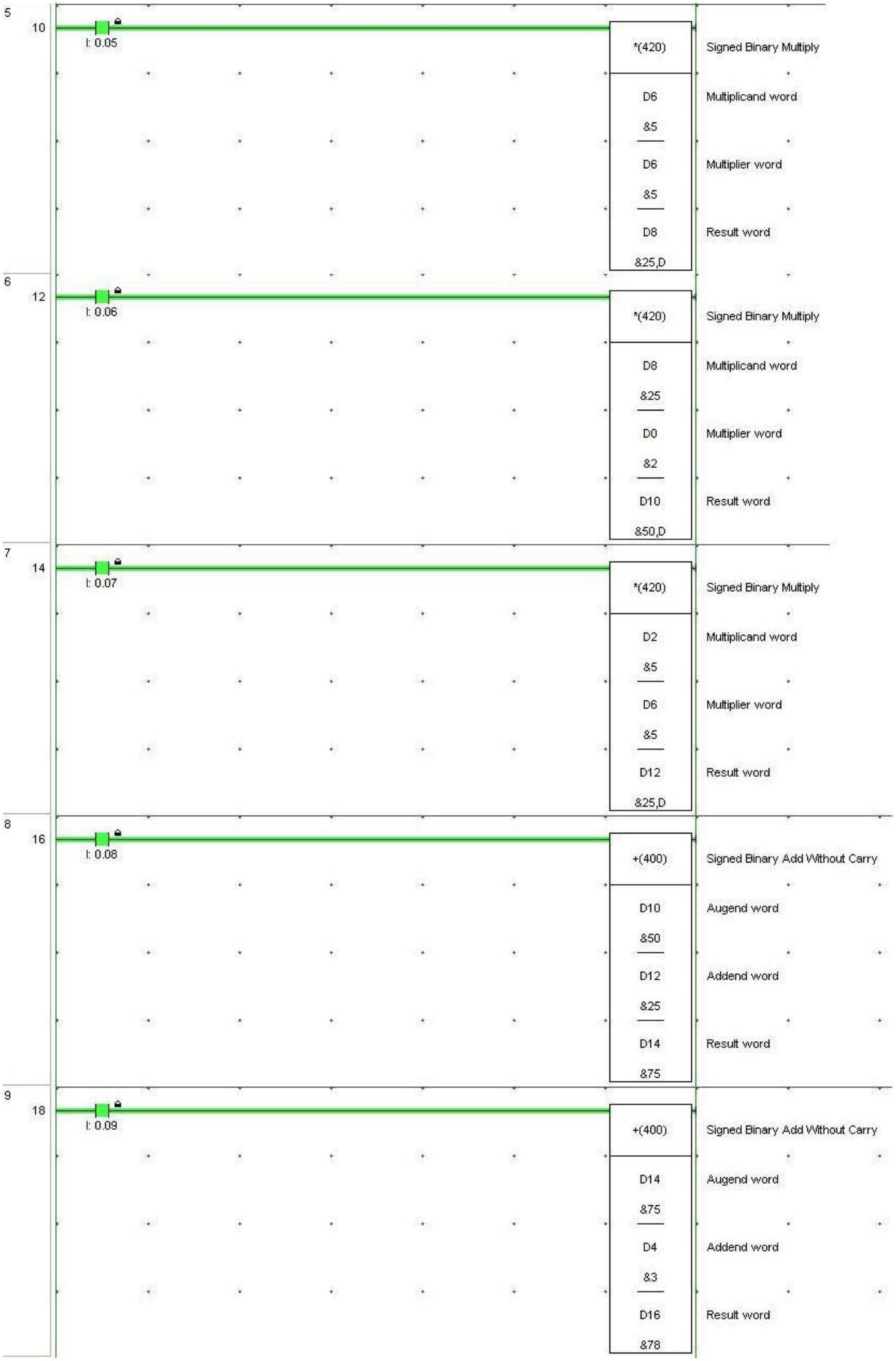
Rys. 2.1. Wyznaczanie wartości funkcji y=ax+b

2.2. Funkcja kwadratowa postaci $y=ax^2+bx+c$

Na rys. 2.2 przedstawiony został program obliczający wartość zadanej funkcji kwadratowej. Współczynniki funkcji zostały przyjęte następująco: a=2 b=5 i c=3. Zmienna x zostaje wprowadzona poprzez komórkę A642. Jest to komórka, której wartość zadajemy pokrętłem analogowym w sterowniku PLC. Jej wartości mogą się zmieniać w przedziale 0-255 ze skokiem co 1. Sygnałem wejściowym 0.00 ustawiona została wartość 0 w rejestrach od D0 do D100 za pomocą instrukcji BSET. W kolejnym kroku wprowadzone

zostały wartości współczynników: a=2, b=5 i c=3 za pomocą funkcji MOV(021) odpowiednio do rejestrów D0, D2, D4. Podobnie jak w poprzednim przykładzie wartość zmiennej x zostaje zadana z pokrętła analogowego sterownika i zapisana w D6. Następnie wykonano iloczyn wartości x w celu uzyskania x w drugiej potędze i zapisania wyniku w D8. Mnożąc rejestry D8 i D0 otrzymuje się pierwszy czynnik trójmianu kwadratowego ax^2. W celu uzyskania iloczynu bx należy pomnożyć rejestry D2 i D6 a wynik tego działania zapisany został w D12. Na końcu zostało tylko sumowanie. Zostało ono zrealizowane w dwóch etapach. Najpierw zsumowano rejestry D10 i D12 zapisując wynik w D14. Ostatnim krokiem jest dodanie do otrzymanego wcześniej wyniku, wartości, która jest zapisana w rejestrze D4. Ostateczny wynik został zapisany w D16.





Rys. 2.2. Wyznaczenie wartości funkcji $y=ax^2+bx+c$