Spesifikasjoner og Sannhetsverditabell

Noen symboler og forkortelser:

- dec: decimal
- bin: binary
- S: fungerer som en 'switch' for multiplikatoren. $S = 0$ hvis $(+10)_{dec}$, og $S = 1$ hvis $(-10)_{dec}$
- T: tallet som skal multipliseres med $(\pm 10)_{dec}$
- F: et positivt binært tall som representeres resultatet til $T \times (+10)_{dec}$
- G: et negativt binært tall som representeres resultatet til $T \times (-10)_{dec}$
- R: multiplikatorens output, d.v.s. $R = F$ hvis $S = 0$, og $R = G$ hvis $S = 1$

Det største tallet som kan representeres med et 4-bit binært tall er $1111_{bin} = 15_{dec}$. Dette vil si at resultatet vi får må ligge mellom -150_{dec} og $+150_{dec}$. Derfor trenger vi minst 9 bits (inkludert sign bit), fordi et 2-er komplement med 9 bits kan representere alle tall fra -256_{dec} til 255_{dec}.

For å kunne multiplisere med både $(+10)_{dec}$ og $(-10)_{dec}$, skal jeg bruke en Switch (S). Dette betyr at multiplikatoren skal få 5 bits som input (en switch og et 4-bit tall), og gi tilbake et 9-bits binært tall som output. Fra disse informasjon, kan vi da sette opp en sannhetsverditabell (se neste side).

Input		Output	Semantikk
S	**T**	**R**	
0	**0000**	**0 0000 0000**	$0 \times 10 = 0$
0	**0001**	**0 0000 1010**	$1 \times 10 = 10$
0	**0010**	**0 0001 0100**	$2 \times 10 = 20$
0	**0011**	**0 0001 1110**	$3 \times 10 = 30$
0	**0100**	**0 0010 1000**	$4 \times 10 = 40$
0	**0101**	**0 0011 0010**	$5 \times 10 = 50$
0	**0110**	**0 0011 1100**	$6 \times 10 = 60$
0	**0111**	**0 0100 0110**	$7 \times 10 = 70$
0	**1000**	**0 0101 0000**	$8 \times 10 = 80$
0	**1001**	**0 0101 1010**	$9 \times 10 = 90$
0	**1010**	**0 0110 0100**	$10 \times 10 = 100$
0	**1011**	**0 0110 1110**	$11 \times 10 = 110$
0	**1100**	**0 0111 1000**	$12 \times 10 = 120$
0	**1101**	**0 1000 0010**	$13 \times 10 = 130$
0	**1110**	**0 1000 1100**	$14 \times 10 = 140$
0	**1111**	**0 1001 0110**	$15 \times 10 = 150$
1	**0000**	**0 0000 0000**	$0 \times (-10) = 0$
1	**0001**	**1 1111 0110**	$1 \times (-10) = -10$
1	**0010**	**1 1110 1100**	$2 \times (-10) = -20$
1	**0011**	**1 1110 0010**	$3 \times (-10) = -30$
1	**0100**	**1 1101 1000**	$4 \times (-10) = -40$
1	**0101**	**1 1100 1110**	$5 \times (-10) = -50$
1	**0110**	**1 1100 0100**	$6 \times (-10) = -60$
1	**0111**	**1 1011 1010**	$7 \times (-10) = -70$
1	**1000**	**1 1011 0000**	$8 \times (-10) = -80$
1	**1001**	**1 1010 0110**	$9 \times (-10) = -90$
1	**1010**	**1 1001 1100**	$10 \times (-10) = -100$
1	**1011**	**1 1001 0010**	$11 \times (-10) = -110$
1	**1100**	**1 1000 1000**	$12 \times (-10) = -120$
1	**1101**	**1 0111 1110**	$13 \times (-10) = -130$
1	**1110**	**1 0111 0100**	$14 \times (-10) = -140$
1	**1111**	**1 0110 1010**	$15 \times (-10) = -150$

Kretsdesign og konstruksjon

La $T = (00000\text{ABCD})_{bin}$ være input til multiplikatoren (hvor ABCD er det originale 4-bits binære tallet). En enkel måte å multiplisere tallet med 2, uten å bruke noen port, er å skifte alle sifrene i tallet én plass til venstre, slik at $T \times 2_{dec} = (0000\text{ABCD}0)_{bin}$. På samme måten, hvis vi mulpliserer T med 8, trenger vi bare å skifte alle sifrene tre plasser til venstre (fordi $2^3 = 8$). Derfor:

$$F = T \times 10_{dec} = (T \times 8_{dec}) + (T \times 2_{dec}) = (00\text{ABCD}000)_{bin} + (0000\text{ABCD}0)_{bin}$$

Nå skal vi bruke 'kolonne' addisjon for å finne hva F må være:

Kolonne		K_8	K_7	K_6	K_5	K_4	K_3	K_2	K_1	K_0
		0	0	A	B	C	D	0	0	0
	+	0	0	0	0	A	B	C	D	0
		F_8	F_7	F_6	F_5	F_4	F_3	F_2	F_1	F_0

La M_i være mente fra kolonne K_i, da har vi følgende beregninger:

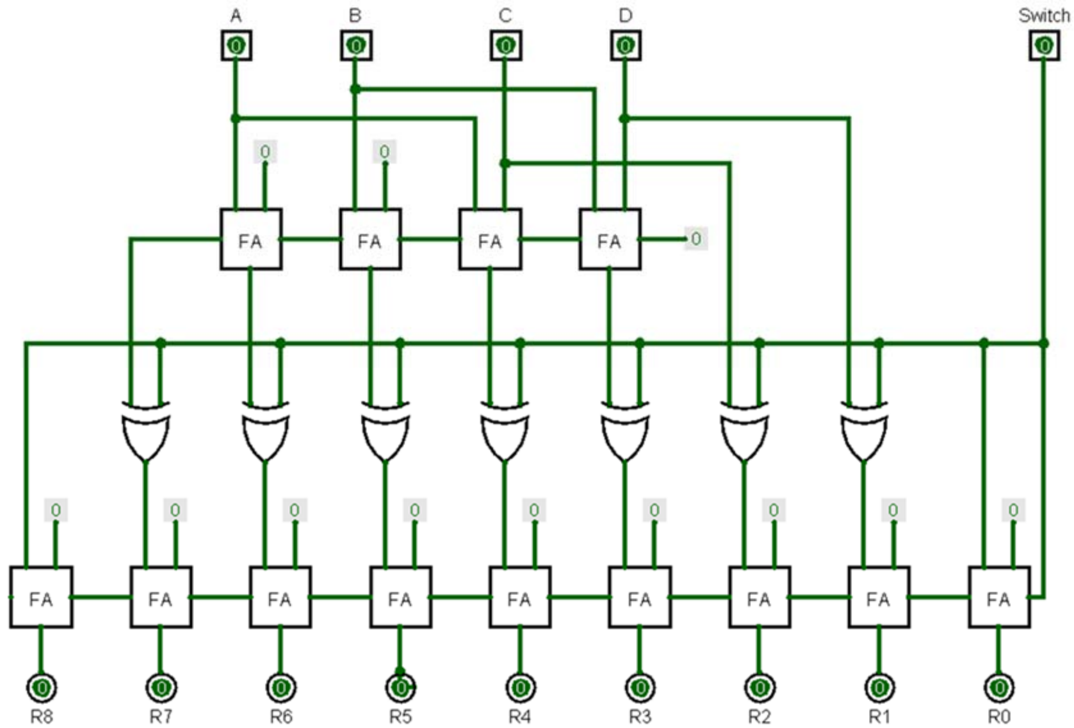
$F_0 = 0$	$F_3 = \mathrm{D} + \mathrm{B}$	$F_6 = \mathrm{A} + \mathrm{M}_5$
$F_1 = \mathrm{D}$	$F_4 = \mathrm{C} + \mathrm{A} + \mathrm{M}_3$	$F_7 = \mathrm{M}_6$
$F_2 = \mathrm{C}$	$F_5 = \mathrm{B} + \mathrm{M}_4$	$F_8 = 0$

➔ Så kan vi bruke full-adder (FA) for bare 4 av de kolonnene, nemlig K_3, K_4, K_5, og K_6.

For å kunne også multiplisere med $(-10)_{dec}$, må vi beregne ut hva det negative tallet G skal bli. Og det enkleste måten å gjøre dette på er å invertere alle sifrene i F, og deretter øke resultatet med 1. Som jeg har nevnt før, skal jeg bruke en switch i kretsen. Hvis $S = 0$ så gjør vi ingenting. Men hvis $S = 1$, må vi invertere alle sifrene F_i. Fra disse informasjon, kan vi sette opp følgende sannhetsverditabell:

Input		**Output**
$\boldsymbol{S}$	$\boldsymbol{F_i}$	
0	0	0
0	1	1
1	0	1
1	1	0

Vi kan få slike tabellen hvis vi bruker XOR port mellom S og hvert siffer i F. Etter at vi har gjort det, kan vi nå bruke et sett av 8 full-adderere for å øke med 1, slik at vi får et negativt tall som resultatet. Ta en titt på den ferdige kretsen på neste side!



Kretsen ovenfor gir riktig resultat, men er litt tungvint fordi kretsen bruker innebygde konstant-bokser i Logisim (dvs. bokser som har forhåndsdefinert sannhetsverdi). Kretsen kan lett forenkles ved hjelp av både half-adder (HA) og full-adder (FA), slik som jeg har gjort for kretsen nedenfor (se også kretsen i Logisim filen).