تنفيذ عمليات الضرب
باستعمال المعالج الدقيق

عبدالناصر يحيى حسين الدكتور رياض كمال الحكيم

جامعة الموصل / كلية الهندسة
قسم الهندسة الكهربائية

الخلاصـــــــــة :

يتناول البحث تنفيذ عمليات الضرب باستعمال المعالج الدقيق (8085) . ولقد اعد هذا البحث بحيث يتلائم مع متطلبات تصميم المرشحات الرقميه والتي تستخدم فيما كسورا " عشرية احادية القطبية وثنائية القطبية . كما يتضمن البحث اعداد جداول مهيئة مسبقا " لغرض الحصول على نتائج الضرب بسرعة . وقد ذكرت الفروق بين هذه الطرق والطرق الاخرى في مناقشة النتائج والاستنتاجات .

1- المقدمـــــــة :

ان الهدف الرئيسي من موضوع البحث هو تصميم المرشحات الرقمية (digital filters) باستعمال المعالج الدقيق (Microprocessor)ونظرا " لظهور الحاجة الى تنفيذ عمليات الضرب فقد تمت دراسة هذه العمليات بالطريقة الملائمة للاعداد والاشارات الكهربائية المستخدمة في معادلات المرشحات الرقمية حيث تكون معظم المعاملات في معادلات المرشحات الرقمية كسورا عشرية (fractions)" . وتكون بعض الاشارات الكهربائية الداخلة على المرشح الرقمي احادية القطبية . كمثال على ذلك الاشارة النبضية (impulse signal) والاشارة التدريجية

(step signal) , والبعض الآخر ثنائية القطبية (اشارة موجبة واشارة سالبة)كما في حالة الموجة الجيبية (sinusoidal wave) تحتوي بعض المعالجات الدقيقة على ايعاز عملية الضرب multiplication instruction مثل (Motorola Mc68000-Intel 8086) ان هذا الايعاز خاص بضرب الاعداد المجردة من الاشارة (unsigned numbers) , ولكن تعويرا "بسيطا " في الناتج يمكننا من استخدام هذا الايعاز لضرب الاعداد ذات الاشارة .

اما غالبية المعالجات الدقيقة فلا تحوي على مثل هذا الايعاز , لذا فأنها تستخدم احدى طريقتين لضرب هذه الاعداد معتمدة على كون الاعداد المضروبة مجردة من الاشارة اومتضمنة لها .

عند استخدام المعالج الدقيق (Intel 8085) لتنفيذ برامج الضرب فأن زمن التنفيذ يعتمد على تردد الساعة (clock frequency) وعدد الايعازات اللازمة .

ان سرعة التنفيذ مهمة جدا " في تطبيقات المعالج الدقيق مع المرشحات الرقمية حيث تتضمن برامج هذه المرشحات تنفيذ اكثر من عملية ضرب لابد ان تنجز انيا " مع خطوات اخرى في فترة زمنية قليلة تعتمد على عرض الحزمة للمرشح . وبما ان تنفيذ عملية الضرب المعدة يستغرق وقتا " طويلا "نسبيا"

فأن استعمال جداول الضرب المعدة مسبقا "(look-up tables) يستغرق زمنا " قليلا " لانجاز عملية الضرب وتخزن نتائج الضرب في مواقع محددة في الذاكرة قبل البدء بتنفيذ البرنامج الرئيس ، واثناء تنفيذ البرنامج لاتحتاج الا لوقت قليل لاخذ النتائج من الجدول بدلا من تنفيذ عملية الضرب بأكملها .

2- طرائق تنفيذ عمليات الضرب

2.1- ضرب الاعداد المجردة من الاشارة (Multiplication of Unsigned numbers) تستخدم هذه الطريقة لضرب الاعداد المجردة من الاشارة في معالجات الاشارة الشفية وغيرها فعملية ضرب القيمة الانية للاشارة الموجبة بعدد موجب تتم بتكرار الجمع . حيث تفحص الارقام الثنائية (binary digits) للضارب (Multiplier) تباعا " من اليمين الى اليسار ، فاذا كان ناتج الرقم مساويا لـ (1) نضيف قيمة المضروب (multiplicand) الى ناتج الضرب (في البداية يكون ناتج الضرب مساويا " للصفر) ، ولايضاف شيئا في حالة كون رقم الضارب صفرا وفي كلتا الحالتين يزحف الناتج مرتبة واحدة الى اليمين ترتيبا للمراتب واستعداد لعملية جمع اخرى . ذكر في المقدمة ان برامج الضرب في هذا البحث معدة

لتصميم المرشحات الرقمية وغالبا مانحتاج في مثـــــــل هذا التصميم الى ضرب اعداد هي عبارة عن قيم الادخال في لحظـــة معينة او الاخراج في لحظة او اخرى باعداد اخرى تمثل معاملات كميات الادخال والاخراج في معاملات المرشحات الرقمية . ان قيم هذه الاعداد هي اقل من الواحد الصحيح ، لذا يكون موقـــــع الفارزة الى اقصى يسار العدد وتمثل الاعداد على النحو الاتي:

$$0.\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1$$
$$2^{-1}\ 2^{-2}\ 2^{-3}\ 2^{-4}\ 2^{-5}\ 2^{-6}\ 2^{-7}\ 2^{-8}$$

اما في حالة كون الاعداد اكبر مـــن الواحد الصحيـــــح فيمكن تمثيل الاعداد بشكل آخر يتلائم مع القيمة العظمى للعدد الصحيح المستخدم ويكون ذلك على حساب الدقة في عدد المراتب الى يمين الفارزة (ذات الموقع الخيالي) .

تمثل عملية الضرب المستخدمة بالمخطط الانسيابي رقـــــــم (1) ، البرنامج الفرعي (subroutine) الخاص بهذه العمليـــــة (MULTIPLY) مكتوب بالشفرة الرمزية (Mnemonic code) للمعالج الدقيق (8085) في ملحق رقم (1) .

يستدعي البرنامج الفرعي (MULTIPLY) برنامجا فرعيا اخر باسم (ROUND) من اجل تقريب ناتج الضرب من (16) مرتبة ثنائيـــة الى ثمانية مراتب ثنائية (اي بايت واحد) ويوضح المخطـــط الانسيابي في الشكل رقم (2) ذلك .

نلاحظ انه في حالة كون الرقم الثنائي ذي المرتبة الاعلى للبايت ذي المرتبة الادنى للناتج مساويا لـ (1) ، فانه يضاف (1) للبايت ذي المرتبة الاعلى من الناتج كتقريب للناتج بطريقة التدوير (ROUNDING) ، خطوات البرنامج الفرعي موجودة في ملحق رقم (2) .

2.2- ضرب الاعداد ذات الرموز ثنائية الاشارة بطريقة (BOOTH)

تستخدم مثل هذه الطريقة عندما تكون الاعداد المضروبة ذات اشارة موجبة وسالبة ، في هذه الحالة تمثل الاعداد على النحو الاتي :

رمز الاشارة ⟵	0.	1	1	1	1	1	1	1
0 للعدد الموجب		2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}
1 للعدد السالب								

ان القيم المطلقة (absolute values) للاعداد الموجبة الممثلة بهذا الرمز هي قيمة العدد نفسه ، اما بالنسبة للاعداد السالبة فأن قيمتها المطلقة هي المتمم لـ 2 (two's complement) للعدد.فمثلا العدد السالب (1.1100000)قيمته المطلقة تعادل 0.0100000.

تستخدم طريقة (BOOTH) لضرب هذه الاعداد بعضها وتتلخص بالخطوات الاتية :

1- افحص التغيير الحاصل في ارقام الضارب من اليمين الى اليسار بين كل رقمين متجاورين مفترضا" صفرا" يمينيا" الى اقصى يمين الضارب .

2- اذا كان الرقمان متشابهين ، انتقل الى الخطوة (5).

3- اذا حصل تغير من (0) الى (1)، اطرح المضروب من ناتج الضرب ثم انتقل الى (5) .

4- اذا كان التغير الحاصل من (1) الى (0) اجمع المضروب مع ناتج الضرب .

5- زحف ناتج الضرب مرتبة واحدة الى اليمين، محافظا على الرقم الثنائي ذي المرتبة الاعلى الموجود في اقصى يسار العدد من دون تغيير .

6- اقصد الخطوة (1) .

المخطط الانسيابي لهذه الطريقة وضع في الشكل رقم (3) .

البرنامج الفرعي الخاص بهذه الطريقة والمسمى (BOOTH) يدون في ملحق رقم (3) .

2.3- جداول الضرب المعدة مسبقا (Multiplication look-up tables)

عند تهيئة جدول الضرب لاحدى المضاعلات مثلا a ، يجب ان نأخذ في نظر الاعتبار ان الكمية المضروبة بـ a هي كلمة من بايت واحد وقد تكون اية قيمة بين (*H 00) و

---

* H ترمز الى نظام hexadecimal 16

(FF H), اي بمعنى انه يجب ان يتسع جدول الضرب لـ (256) موقع في الذاكرة لخزن نتائج ضرب a بالقيم من (00H) الى (FF H) النتائج تخزن في مواقع ذاكرة تتراوح عناوينها مثلا من (8100 H) الى (81FF H) .

اما طريقة الحصول على نتائج الضرب بعد معرفة الكمية المراد ضربها بـ a ,تتلخص بوضع هذه الكمية في السجل (A)في المعالج الدقيق , حيث تضاف الى العنوان الاول لجدول الضرب الخاص بالمعامل a وهو (8100 H) ناتج الجمع يشير الى موقع الذاكرة التي تحوي ناتج الضرب .

شكل رقم (4) يوضح المخطط الانسيابي الخاص بتهيئة جدولي ضرب لمعاملين هما مثلا (a,b) الجدول الاول الخاص بـ a يتراوح من عنوان ذاكرة (8100 H) الى (81FF H) الجدول الثاني خاص بـ b يتراوح من عنوان الذاكرة (8200 H)والى (82FF H).

البرنامج الخاص بتهيئة هذين الجدولين (PLUT routine) موجود في ملحق رقم (4). هذا البرنامج يستدعي برنامجا "فرعيا" باسم (TABLE) المخطط الانسيابي لهذا البرنامج الفرعي موجود في شكل رقم (5) وخطوات البرنامج موجودة في الملحق رقم (5) .

3- النتائـــــــــج :

1- في حالة ضرب الاعداد المجردة من الاشارة ، يستغـــــــــرق البرنامج الفرعي (MULTIPLY)زمنا قدره (0.22135 ms). اما البرنامج الفرعي (ROUND) يستغـرق (17.496 MS) ، وكلاهما معا"يستغرقان زمنا"مقـــــــداره (0.2388505 ms) اعتمادا على كون تردد الساعة في المعالج الدقيـق (8085) المستخدم مساويا لـ (2.4576 MHZ) . الزمن الذي يستغرقـه المعالج الدقيق لغرض تنفيذ البرنامج الفرعـي (BOOTH) يتراوح بين اقل زمن (0.3169759 ms) عندما يكون الضارب (00H). واطول زمن مقداره (0.39835612 ms) عندمـــا يكون الضارب (55H) (01010101) الذي يحوي اكبـــر عدد من التغييرات في الارقام من 0 الى 1 وبالعكس .

2- في البرنامج الفرعي (MULTIPLY)تمثل الاعداد بدقة 2^{-8} (قيمة الرقم الثنائي ذي المرتبة الادنى) ، امـــــا في البرنامج الفرعي (BOOTH)تمثل الاعداد بدقة 2^{-7} (قيمة الرقم الثنائي ذي المرتبة الادنى) بسبب استخدام الرقم الثنائي ذي المرتبة الاعلى للتعبير عن اشارة العـــدد في البرنامج الفرعي (BOOTH) .

3- الزمن اللازم للحصول على ناتج الضرب من جدول الضـــرب المهيئ مسبقا يساوي (10.172525 MS) .

4- الاستنتاجـــــات

1- استخدام جداول الضرب يوفر الوقت ولكنه يحتاج الـــى توفير اماكن حجز في ذاكرة الحاسبة ، حيث يحتـــاج كل جدول ضرب الى(256)موقعا في الذاكرة لخزن نتائج الضرب كلما كان الزمن هو العامل المهم في التصميـــم فأن جداول الضرب مفضلة كثيرا على استدعاء البرامـــج الفرعية للضرب ، هذا بشرط توفر مواقع كافيـــة في الذاكرة . وكلما كان الزمن اللازم لاكمال البرنامـــج طويلا "نسبيا "قياسا" الى زمن تنفيذ برامج الضرب فأن عملية استدعاء برامج الضرب تصبح ملائمة ومفضلة علـــى طريقة الجداول لعدم حاجتها الى توفر مواقع الحجز في الذاكرة اللازمة لجداول الضرب ، وبهذا توفر مجـــالا "كبيرا"في الذاكرة لغرض استخدامها في اغراض اخرى .

2- هنالك بعض الحاسبات الدقيقة تحوي على سجـــــل هو (index register) مثل (Motorola Mc6800) فائدة هذا السجل هو اختصار زمن الحصول على النـاتـــج من جدول الضرب العمليه مسبقا " يخزن مسبقا" في هذا السجـــل العنوان الاول لجدول الضرب ومن ثم يضاف اليه الكميـة المراد ضربها بالمعامل العمليه له جدول الضرب فيكـون الناتج هو عنوان الذاكرة التي تحوي ناتج الضرب .

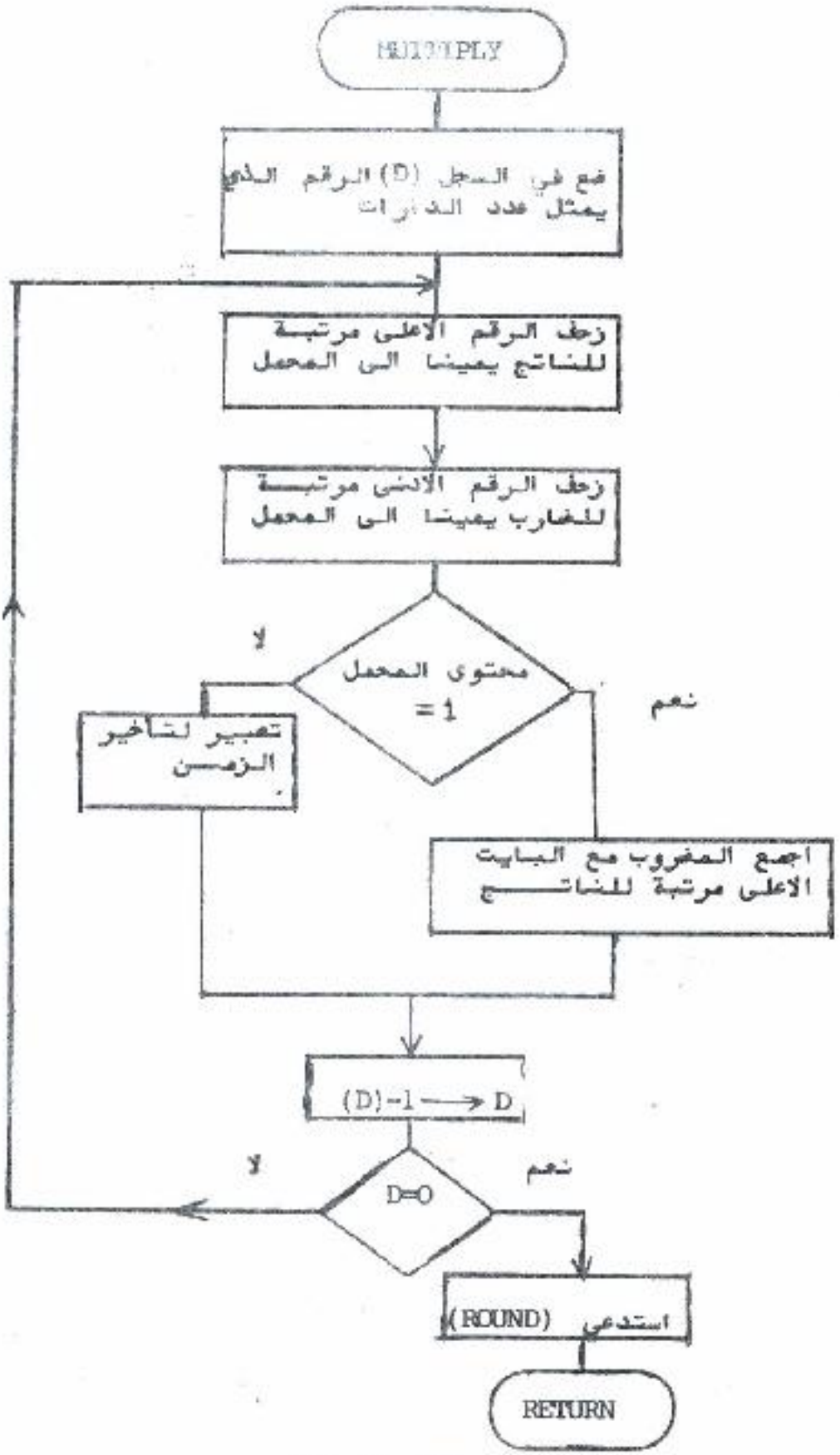
3- تتناول هذا البحث طريقتين للضرب باستخدام البرنامجين الفرعيين (MULTIPLY) و (BOOTH), الثاني اكثر شمولا بسبب صلاحيته للاعداد التي تكون رموزها حاوية على اشارة موجبة واشارة سالبة على الرقم من ان البرنامج الفرعي الاول اقصر في زمن التنفيذ كما ملاحظ في النتائج , وكذلك اكثر دقة في تمثيل الاعداد من البرنامج الفرعي الثاني (BOOTH) .

4- ان زمن تنفيذ البرامج الفرعية للضرب يعتمد على تردد الساعة للمعالج الدقيق وباستخدام معالجات دقيقة ذات ترددات ساعة اعلى يمكن زيادة سرعة التنفيذ مثال على ذلك المعالج الدقيق (8085 A -2) .

5- ان الزمن اللازم للحصول على ناتج الضرب من جدول الضرب المهيئ مسبقا يساوي (10.172525 MS) . ان استخدام (hardware multiplier) يستغرق الضرب فيه زمنا مقداره (100 MS) وبهذه الحالة نكون قد وفرنا زمنا في التنفيذ على حساب الكلفة .

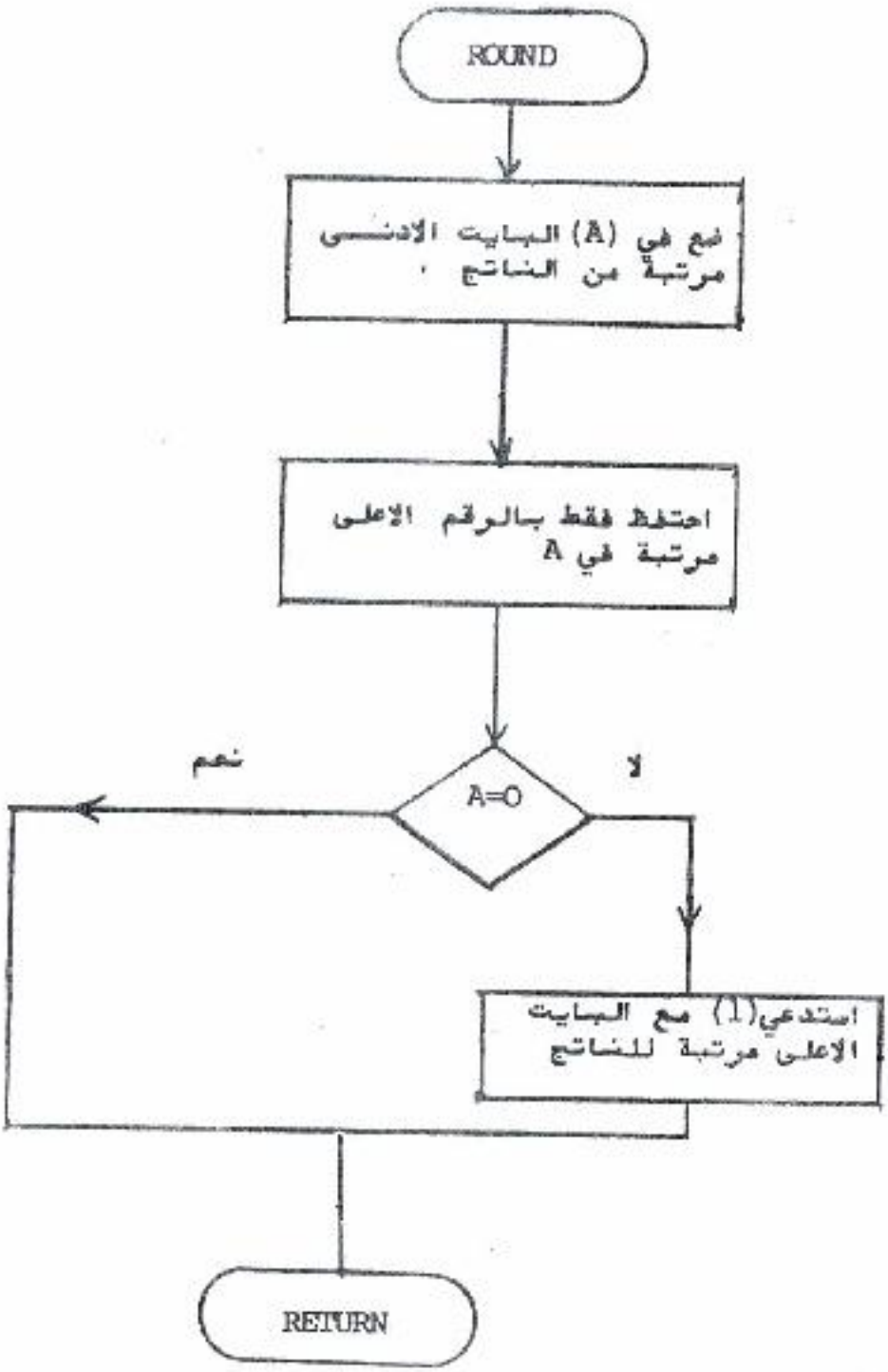
5- المصادر

MOTOROLA;Microprocessor Application Manual"Mcgrawhill, 1975 .

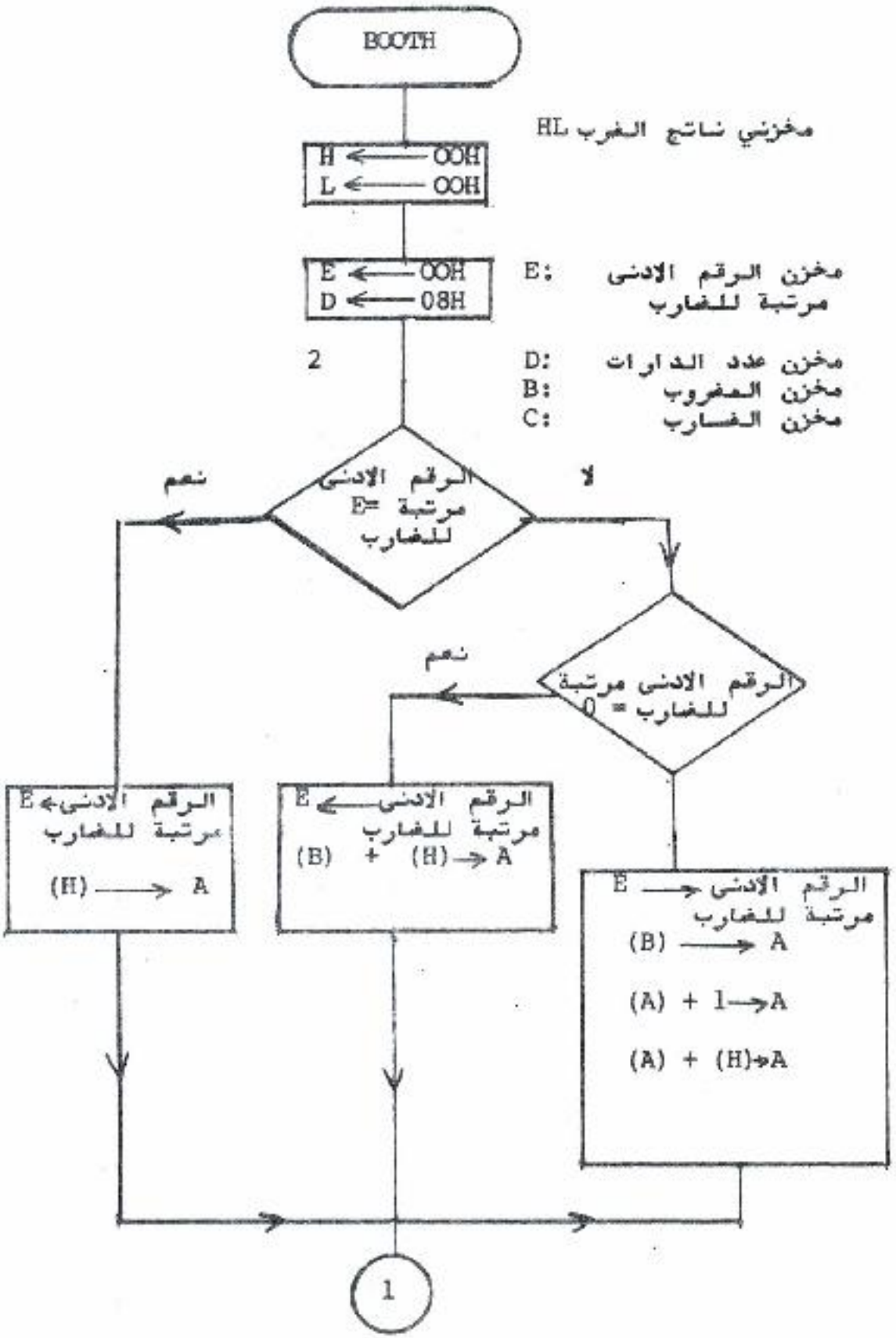


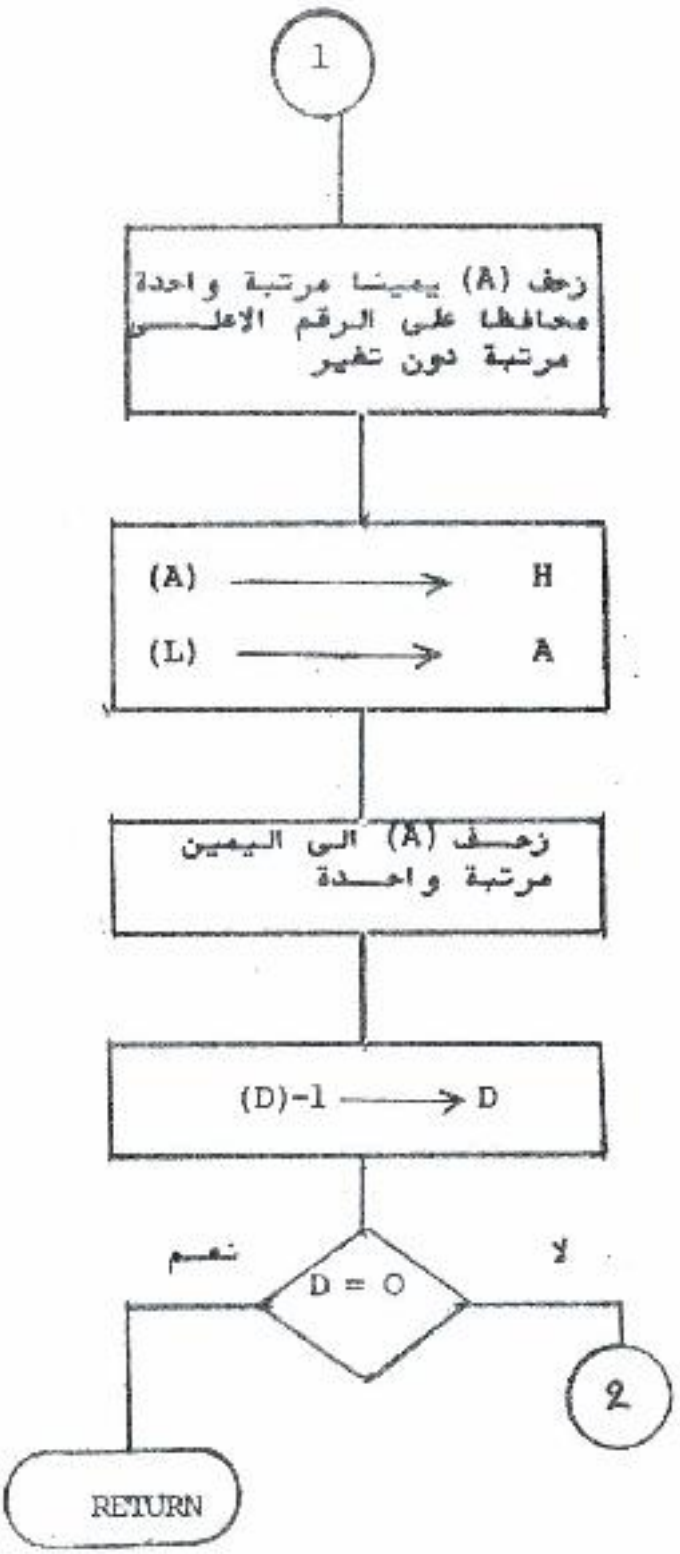
شكل رقم (1)

المخطط الانسيابي للبرنامج الفرعي ((MULTIPLY))

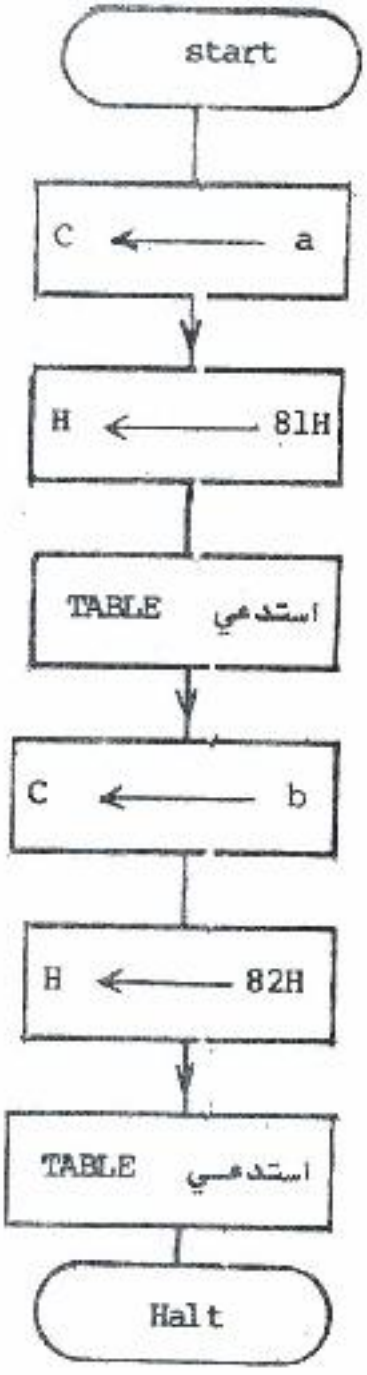


شكل رقم (2)
المخطط الانسيابي للبرنامج الفرعي (ROUND)

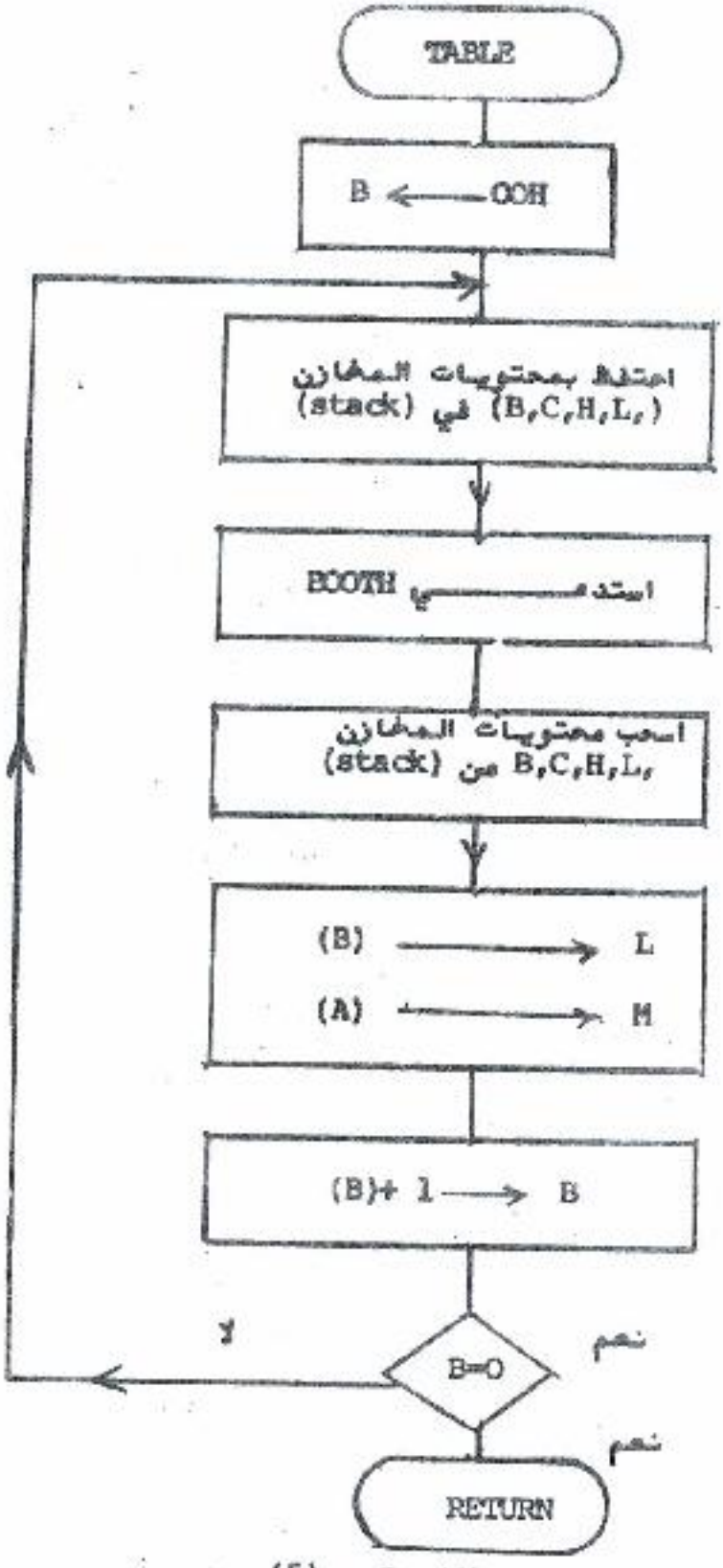

BOOTH
مخزني ناتج الضرب HL
H ← 00H
L ← 00H
E ← 00H
D ← 08H
E: مخزن الرقم الادنى مرتبة للمضارب
2
D: مخزن عدد الدورات
B: مخزن المضروب
C: مخزن المضارب
الرقم الادنى مرتبة E= للمضارب
نعم
لا
الرقم الادنى مرتبة للمضارب = 0
نعم
الرقم الادنى ← E مرتبة للمضارب
(H) → A
الرقم الادنى ← E مرتبة للمضارب
(B) + (H) → A
الرقم الادنى ← E مرتبة للمضارب
(B) → A
(A) + 1 → A
(A) + (H) → A
1



شكل رقم (3)
المخطط الانسيابي للبرنامج الفرعي (BOOTH)



شكل رقم (4)
المخطط الانسيابي لبرنامج (Plut)



شكل رقم (5)

المخطط الانسيابي للبرنامج الفرعي TABLE

ملحق رقم (1)

البرنامج الفرعي (MULTIPLY)

Lable	Mnemonic
MULTIPLY	MVI D.09
	MOV B,A
	XRA A
LA2	RAR
	MOV E,A
	MOV A,C
	RAR
	MOV C,A
	MOV A,E
	JNC NA2
	ADD B
	MVI H,00
	DCR D
	JNZ LA2
	CALL ROUND
	RET
NA2	NOP
	NOP
	DCR D
	JNZ LA2
	CALL ROUND
	RET

ملحق رقم (2)

البرنامج الفرعي (ROUND)

Lable	Mnemonic	
ROUND	MOV	E,A
	MOV	A,C
	ANI	80
	JZ	NOADD
	MOV	A,E
	ADI	01
	RET	
NOADD	MOV	A,E
	NOP	
	RET	

ملحق رقم (3)

البرنامج الفرعي (BOOTH)

Lable	Mnemonic		Lable	Mnemonic	
BOOTH	LXI	H,0000	NO	MOV	E,A
	LXI	D,0000		MOV	A,H
	MOV	B,A	SHIFT	RIC	
LAI	XRA	A		RAR	
	MOV	A,C		RAR	
	RAR			MOV	H,A
	MOV	C,A		MOV	A,L
	RAL			RAR	
	ANI	OI		MOV	L,A
	CMP	E		DCR	D
	JZ	EO		JNZ	LAI
	JC	ADD		XRA	A
SUB	MOV	E,A		MOV	A,L
	MOV	A,B		RAL	
	CMA			MOV	A,H
	ADI	OI		RAL	
	ADD	H		RET	
	JMP	SHIFT			
ADD	MOV	E,A			
	MOV	A,H			
	ADD	B			
	JMP	SHIFT			

ملحق رقم (4)

البرنامج الفرعي (PLUT)

Lable	Mnemonic	
	MVI	C,a
	MVI	H,81
	CALL	TABLE
	MVI	C,b
	MVI	H,82
	CALL	TABLE
	HALT	

ملحق رقم (5)

البرنامج الفرعي (TABLE)

Lable	Mnemonic		Lable	Mnemonic	
TABLE	MVI	B,00		MVI	A,00
LA4	Push	H		CMA	B
	Push	B			
	CALL	BOOTH		JNZ	LA4
	POP	B		RET	
	POP	H			
	MOV	L,B			
	MOV	M,A			
	INR	B			