

5 الفصل الخامس

برمجة المعالج Z80

Programming The Z80 Microprocessor

5-1 مقدمة

يعتبر المعالج Z80 صورة متطورة ومنقحة للمعالج Intel8085 حيث أن جميع أوامر الشريحة 8085 تتوافق مع الشريحة Z80 ولكن الشريحة Z80 تمتاز ببعض المميزات الأخرى الغير موجودة في الشريحة 8085 والتي سنراها في أثناء دراستنا لهذا الجزء . سنتبع في هذا الفصل نفس الطريقة التي اتبعناها في الفصل السابق حيث سنقسم أوامر الشريحة Z80 إلى مجموعات وسندرس بالتفصيل من كل مجموعة بعض الأوامر الكثيرة الاستخدام على أننا سنعرض في نهاية الفصل لجدول مختلفة لجميع الأوامر حيث يمكن للقارئ الرجوع إليها عند الحاجة . ولقد تعمدنا أن نتبع نفس طريقة الشرح في الفصل السابق حتى يتمكن من يريد أن يتتبع المعالج Z80 فقط دون اللجوء إلى مراجعة أى معالج آخر قد لا يحتاج إليه في أثناء تدريبيه ، كما أننا سنعرض مقارنات بسيطة بين المعالج Z80 والمعالج 8085 في بعض المواقف التي تتطلب ذلك حتى تكتمل الفائدة لمن يريد ذلك .

5-2 مجموعة أوامر الانتقال

Transfer instructions

يقوم أى أمر من أوامر هذه المجموعة بنقل معلومة (بايت) من مكان لآخر حيث المكان الذى تخرج منه المعلومة سنسميه المصدر Source وسنرمز له أحيانا بالرمز sss وهذا المكان قد يكون مسجلا داخل شريحة المعالج وقد يكون بايتا من بايتات الذاكرة ، وأما المكان الذى ستذهب إليه المعلومة فسوف نسميه الهدف Destination وسنرمز له أحيانا بالرمز ddd وهذا المكان أيضا قد يكون مسجلا داخل شريحة المعالج وقد يكون بايتا من بايتات الذاكرة كما سنرى . تمتاز جميع أوامر الانتقال الخاصة بالشريحة Z80 بأن لها نفس الأحرف مهما كان مصدر أو هدف المعلومة وهذه الأحرف هي LD التي هي اختصار لكلمة LOAD أو حمل ، وذلك على العكس من الشريحة 8085 التي تستخدم عددا أكثر من الأحرف والتي منها MOV و MVI و LDA وغير ذلك من الصور التي تستخدم كل منها في حالته الخاصة كما رأينا في الفصل السابق لذلك سنتناول أوامر الانتقال في حالة الشريحة Z80 على حسب مصدر وهدف المعلومة كما يلي :

5-2-1 نقل معلومة من مسجل الى مسجل آخر

الصورة العامة لهذا الأمر هي :

LD ddd,sss

مسجل sss ← مسجل ddd

ومعناه تحميل المسجل ddd بمحتويات المسجل sss ، لاحظ أن الذي يتم نقله من المسجل sss هو صورة من المحتويات فقط أما محتويات المسجل فتظل كما هي لا تتغير . من الأمثلة على ذلك ما يلي :

- الأمر LD A,B حيث سيقوم هذا الأمر بنقل (نسخ) محتويات المسجل B (المصدر) إلى المسجل A (الهدف) .
- الأمر LD C,H سيقوم بتحميل المسجل C بمحتويات المسجل H .
- الأمر LD B,B سيقوم بتحميل المسجل B بمحتويات المسجل B . لاحظ أن تأثير هذا الأمر يكافئ تماما "لا تعمل شيئا" وهذه العملية تكون مهمة جدا في الكثير من التطبيقات ولذلك فقد أفرد لها أمر خاص بها وهو الأمر NOP أى No Operation أو لا تعمل شيئا ، وهذا الأمر سنراه في الكثير من التطبيقات القادمة إن شاء الله . الشفرات الست عشرية لجميع أوامر الانتقال بين جميع المسجلات بعضها البعض يوضحها شكل (5-1) . بالنظر لهذا الشكل نجد أنه إذا أردنا مثلا نقل محتويات المسجل A إلى المسجل L نستخدم الأمر LD L,A الذي شفرته من شكل (5-1) هي 6F . جميع أوامر نقل المعلومة من مسجل إلى آخر تتكون من بايت واحدة فقط تسمى OP Code وهي اختصار Operation Code أو شفرة العملية .

5-2-2 تحميل مسجل بمعلومة فورية أو ثابت Constant

في الكثير من التطبيقات نحتاج لتحميل مسجل من المسجلات بثابت معين ، في هذه الحالة تكون الصورة العامة لمثل هذه الأوامر كما يلي :

LD ddd,data8

data8 ← المسجل ddd

ومعناه حمل المسجل ddd بالمعلومة الفورية أو الثابت data8 ، لاحظ أن data8 يقصد بها ثابت مكون من ثمانية بتات ، جميع أوامر هذه المجموعة تتكون من اثنين بايت واحدة هي شفرة الأمر Op Code والثانية هي البايث data8 أو الثابت. شكل (5-2) يبين الشفرات الست عشرية لعملية تحميل أى مسجل من المسجلات بثابت data8 . من هذا الشكل نلاحظ أنه لتحميل المسجل D بالقيمة 55H مثلا نستخدم الأمر LD D,55H والذي ستكون شفرته الست عشرية كما يلي :

16

55H

حيث 16 (البايت الأولى) هي op code كما في شكل (2-5) أما البايت الثانية فهي قيمة الثابت المراد تحميله في المسجل D وهو 55H ، لاحظ أن H بعد الرقم تعنى أن هذا الرقم ممثل في النظام الستعشرى .

مسجل الهدف Destination register	مسجل المصدر Source register						
	A	B	C	D	E	H	L
A,	7F	78	79	7A	7B	7C	7D
B,	47	40	41	42	43	44	45
C,	4F	48	49	4A	4B	4C	4D
D,	57	50	51	52	53	54	55
E,	5F	58	59	5A	5B	5C	5D
H,	67	60	61	62	63	64	65
L,	6F	68	69	6A	6B	6C	6D

شكل (1-5) الشفرة الستعشرية لجميع أوامر الإنتقال بين المسجلات

A	B	C	D	E	H	L
3E	06	0E	16	1E	26	2E
data8						

شكل (2-5) الشفرات الستعشرية لأوامر تحميل المسجلات بمعلومة فورية أو ثابت data8

مثال 1-5

المطلوب تحميل المسجلات A, B, C, D, E, H بالمعلومات الفورية أو الثوابت الآتية : 01, 02, 03, 04, 05, 06 وبعد ذلك يتم إجراء إزاحة دورانية لهذه المحتويات بحيث تذهب محتويات المسجل A إلى المسجل B وتذهب محتويات المسجل B إلى المسجل C وهكذا إلى أن نصل إلى محتويات المسجل H التي تذهب إلى المسجل A . شكل (2-4) في الفصل السابق يبين رسماً توضيحياً ومخطط السير لهذا البرنامج ، أما البرنامج بلغة الأسمبلى والشفرات الستعشرية فإنه موضح في شكل (3-5) . شكل (3-4) في الفصل السابق يبين نفس هذا البرنامج مكتوباً بلغة الأسمبلى الخاصة بالشريحة 8085 وبمقارنة الشفرات الستعشرية لكلا البرنامجين في الشكلين (3-5 و 3-4) نجد أن هناك تطابقاً تاماً في الشفرات الستعشرية في الحالتين وهذا يبين مدى التطابق بين الشريحتين Z80

و 8085 فإن أى برنامج مكتوبا بالشفرات الست عشرية للشريحة 8085 يمكن تنفيذه باستخدام الشريحة Z80 وأما البرامج المكتوبة بالشفرات الست عشرية للشريحة Z80 فليس بالضرورة أنها يمكن تنفيذها على الشريحة 8085 وذلك لأن الشريحة Z80 تحتوى على عدد أكثر من الأوامر الغير معرفة لدى الشريحة 8085 .

العناوين	شفرات أسمبلى	شفرات ست عشرية
E000	LD A,01	3E
		01
E002	LD B,02	06
		02
E004	LD C,03	0E
		03
E006	LD D,04	16
		04
E008	LD E,05	1E
		05
E00A	LD H,06	26
		06
E00C	LD L,A	6F
E00D	LD A,H	7C
E00E	LD H,E	63
E00F	LD E,D	5A
E010	LD D,C	51
E011	LD C,B	48
E012	LD B,L	45
E013	HLT	76

شكل (3-5) برنامج الأراحة الدورانية

لتنفيذ البرنامج الموجود فى شكل (3-5) فإنه يمكن أن ندخل فى الذاكرة RAM ونكتب الشفرات الست عشرية ابتداء من العنوان E000 وبعد الانتهاء من كتابة البرنامج فى الذاكرة ننفذه باستخدام الأمر GO E000 . أما إذا كان الميكروكومبيوتر الذى نستخدمه به الأسمبلر الخاص بالمعالج Z80 فإنه يمكننا فى هذه الحالة كتابة البرنامج بلغة الأسمبلى مباشرة ثم تنفيذه وسنترك تفاصيل عملية إدخال البرنامج لأنها تختلف من شخص لآخر ولكننا ننصح أن تكتب

البرامج الأولى في مرحلة التدريب بالشفرات الست عشرية ثم بعد ذلك تكتب بلغة الأسمبلى ونصح أيضا أن يتم تنفيذ البرامج الأولى بطريقة الخطوة خطوة حتى يتمكن المتدرب من ملاحظة تأثير كل أمر على حده ومتابعة تحميل المسجلات وانتقال البيانات من مسجل لآخر .

يمكن أيضا تحميل زوج من المسجلات بمعلومة مكونة من 16 بتا كما يلي :

LD rp,data16

rp ← data16 زوج المسجلات

حيث rp ترمز لأي زوج من أزواج المسجلات المتاحة في المعالج Z80 وهي BC, DE, HL, SP, IX, IY (راجع شكل (2-6) لترى أزواج المسجلات المتاحة في الشريحة Z80) . شكل (5-4) يبين الشفرات الست عشرية اللازمة لتحميل كل زوج من أزواج هذا المعالج . لاحظ أن جميع هذه الأوامر ستتكون من ثلاث بايتات ، واحدة (الأولى) ستكون شفرة الأمر op code والاثنتان التاليتان ستحتويان المعلومة data16 المكونة من 16 بت كما ذكرنا فيما عدا في حالة الزوج IX و IY فإن الأمر سيتكون من أربعة بايتات ، إثنان لشفرة الأمر op code وإثنان للمعلومة data16 . لاحظ أننا مجازا فقط في هذا المكان نطلق على المسجلات BC, IX, SP كلمة أزواج ولكن ليكن راسخا في العلم أن كل واحدة منها عبارة عن مسجل واحد مكون من 16 بت ولا يمكن تقسيمه إلى مسجلين . كما هو الحال في الزوج BC مثلا الذي يمكن استخدامه كمسجلين B و C . كمثل على ذلك فإن الأمر :

BC	DE	HL	SP	IX	IY
01	11	21	31	DD21	FD21
data16	data16	data16	data16	data16	data16

شكل (5-4) الشفرات الست عشرية لأوامر تحميل أزواج المسجلات بمعلومة فورية أو ثابت data16

LD HL,E100

سيحمل الزوج HL بالمعلومة E100H حيث ستذهب البايث الأولى من المعلومة (00) إلى المسجل L والبايث الثانية من المعلومة (E1) ستذهب إلى المسجل H . الشفرات الست عشرية لهذا الأمر ستكون كالتالي :

21
00
E1

5-2-3 نقل معلومة من مسجل الى الذاكرة والعكس

نقل معلومة من مسجل في داخل المعالج Z80 إلى أى مكان في الذاكرة أو العكس يمكن استخدام طريقة من ثلاث طرق متاحة لهذا الغرض وهى كالتالى :

5-2-3-1 الطريقة المباشرة Direct addressing

في هذه الطريقة يكون عنوان الباييت المراد التعامل معها موجودا في الأمر نفسه (في الباييت الثانية والثالثة) ولذلك فإن مثل هذه الأوامر تتكون دائما من ثلاث بايتات واحدة هي شفرة الأمر op code واثنان للعنوان المراد التعامل معه . هناك أمران فقط تحت هذه المجموعة ، أمر خاص بنقل المعلومات من مسجل التراكم إلى الذاكرة والآخر خاص بنقل المعلومات من الذاكرة إلى مسجل التراكم A ولذلك فاننا نلاحظ أن التعامل بالطريقة المباشرة لا يكون إلا بين مسجل التراكم فقط والذاكرة ، فإذا أردنا نقل معلومة مثلا من المسجل B إلى الذاكرة بهذه الطريقة فإننا ننقل المعلومة أولا إلى المسجل A ثم منه إلى أى مكان في الذاكرة . الأمر الأول في هذه المجموعة هو :

LD A,(addr)

A ← (addr) المسجل

الشفرة الست عشرية لهذا الأمر هي :

3A

البايت ذات القيمة الصغرى من العنوان addr

البايت ذات القيمة العظمى من العنوان addr

حيث سيقوم هذا الأمر بنقل محتويات باييت الذاكرة التى عنوانها في الاثنتين باييت الثانية والثالثة في الأمر نفسه إلى مسجل التراكم A . ونؤكد هنا على كلمة محتويات حتى لا يظن البعض أن العنوان نفسه هو الذى يوضع في المسجل A كما يوحي شكل الأمر لأول وهلة ولقد تم وضع قوسين حول كلمة addr في الصورة الحرفية للأمر للتأكيد على ذلك ولأنه بدون هذين القوسين لن يستطيع الأسمبلر التمييز بين ما إذا كان الرقم addr عنوانا أم معلومة فورية مطلوب تحميلها في المسجل A . كمثال على ذلك انظر إلى الأمر :

LD A,(E100)

حيث سيقوم هذا الأمر بتحميل المسجل A بمحتويات العنوان E100 .

الأمر الثانى من أوامر هذه المجموعة هو :

LD (addr),A

(addr) ← A المسجل

والشفرة الست عشرية لهذا الأمر هي :

32

البايت ذات القيمة الصغرى من العنوان addr

البايت ذات القيمة العظمى من العنوان addr

حيث سيقوم هذا الأمر بنقل محتويات المسجل A إلى بايت الذاكرة التي عنوانها موجود في البايت الثانية والثالثة من الأمر نفسه ، أى أن هذا الأمر يقوم بالعملية العكسية للأمر السابق . كمثال على ذلك انظر إلى الأمر :

LD (E100),A

والذى شفرته الست عشرية ستكون :

32

00

E1

حيث سيقوم هذا الأمر بتخزين محتويات المسجل A فى بايت الذاكرة التى عنوانها E100 . بذلك نكون قد انتهينا من الطريقة المباشرة لعنونة أو التعامل مع الذاكرة . إن هذه الطريقة كما رأينا مثلها مثل أن تقول لصديقك أحمد ياأخى ياأحمد وأنت مسافر إلى الرياض أرجوك أن تعطى هذا الخطاب لأخى محمد هناك وعنوانه موجود على الخطاب مباشرة ، هنا صديقك أحمد الذى يمثل المعالج الذى سينفذ الأمر سيعرف عنوان أخيك فى الرياض من على الخطاب مباشرة حيث الخطاب فى هذه الحالة يعتبر الأمر المطلوب تنفيذه .

2-3-2-5 الطريقة غير المباشرة Indirect addressing

فى هذه الطريقة لا يكون عنوان البايت المراد التعامل معها فى الذاكرة موجودا مباشرة فى الأمر نفسه ولكنه يكون موجودا فى مكان آخر وعلى المعالج الذهاب إلى هذا المكان أولا وقبل تنفيذ الأمر لمعرفة العنوان . هذا المكان يكون زوجا من المسجلات ولا بد أن يكون زوجا لأن العنوان كما نعرف دائما يتكون من 16 بت ، وعادة يكون هذا الزوج هو الزوج HL . الصورة العامة للأمر فى هذه الحالة بلغة الأسمبلى تكون كما يلى :

LD ddd,(HL)

ddd ← (HL) المسجل

حيث سيقوم هذا الأمر بنقل محتويات البايت التى عنوانها فى زوج المسجلات HL إلى المسجل ddd . الصورة العكسية لهذا الأمر هي :

LD (HL),sss

(HL) ← sss المسجل

حيث سيقوم هذا الأمر بنقل محتويات مسجل المصدر إلى بايت الذاكرة التي يوجد عنوانها في زوج المسجلات HL . شكل (5-5) يبين الشفرات الست عشرية لهذه الأوامر في حالة استعماله مع جميع المسجلات الموجودة في الشريحة Z80 وكما ذكرنا فإن العنوان لا بد وأن يكون موجودا في الزوج HL . هذه الأوامر (أوامر الطريقة غير المباشرة) ستكون جميعها مكونة من بايت واحدة فقط وهي شفرة الأمر op code . كمثال على ذلك الأمر LD B,(HL) والذي ستكون شفرته الست عشرية 46 حيث سيقوم هذا الأمر بنقل محتويات الباييت التي عنوانها في الزوج HL إلى المسجل B . لاحظ وجود القوسين حول الزوج HL في جميع هذه الأوامر للدلالة على أن المقصود هو محتويات العنوان الموجود في HL وليس القيمة الموجودة في HL مباشرة .

	A	B	C	D	E	H	L
ddd ← (HL)	7E	46	4E	56	5E	66	6E
(HL) ← sss	77	70	71	72	73	74	75

شكل (5-5) الشفرات الست عشرية لأوامر الانتقال بين المسجلات والذاكرة بالطريقة غير المباشرة

هناك ميزة يمتاز بها المسجل A عن باقى مسجلات المعالج في حالة التعامل بينه وبين الذاكرة بالطريقة غير المباشرة وهي أن عنوان الباييت المراد التعامل معها في هذه الحالة يمكن وضعه في أى زوج من أزواج المسجلات الأخرى وليس بالضرورة الزوج HL كما سبق ، وذلك كما قلنا كحالة خاصة فقط للمسجل A . شكل (5-6) يبين الشفرة الست عشرية والأسمبلى وما يقوم به كل واحد من هذه الأوامر .

الشفرة الأسمبلى	الشفرة الست عشرية	ما يفعله الأمر
LD A,(BC)	0A	A ← (BC)
LD A,(DE)	1A	A ← (DE)
LD (BC),A	02	(BC) ← A
LD (DE),A	12	(DE) ← A

شكل (5-6) أوامر الانتقال بين المسجل A والذاكرة بالطريقة غير المباشرة

كما رأينا فإن الطريقة غير المباشرة في التعامل مع الذاكرة مثلها مثل أن تقول لصديقك أحمد ياأخي ياأحمد وأنت مسافر إلى الرياض خذ هذا الخطاب وأعطه لأخي محمد ولكن أرجوك قبل سفرك أن تمر على والدى لتعرف منه عنوان

أخي محمد في الرياض لأنى لا أعرفه . هنا صديقك أحمد يمثل المعالج الذى سيقوم بالتنفيذ والوالد يمثل المسجلين HL حيث يحتويان العنوان المراد التعامل معه واللذان لابد من المرور عليهما قبل تنفيذ الأمر . السؤال الآن كيف نضع عنوان ما في زوج من المسجلات ؟ إن هذه العملية بسيطة جدا حيث يستخدم فيها أوامر تحميل زوج مسجلات بمعلومة فورية أو ثابت مكون من 16 بت والتي درسناها في الجزء السابق (5-2-2 تحميل مسجل بمعلومة فورية) .

5-2-3 طريقة الفهرسة لعنونة الذاكرة Indexed addressing

هناك مسجلان لم نتكلم عنهما تفصيليا إلى الآن وهما المسجلان IX و IY وكل منهما يتكون من 16 بت . هذان المسجلان يختلفان عن أزواج المسجلات الأخرى فى أنه لا يمكن استخدام أى منهما كمسجلين 8 بتات منفصلين ولكن كل منهما يستخدم كمسجل 16 بت مثله فى ذلك مثل عداد البرنامج program counter أو مؤشر المكعدة stack pointer . هذان المسجلان يستخدمان فى عملية الإشارة إلى بايتات الذاكرة تماما مثل الزوج HL ولكن بإمكانيات أكثر ، ولكى نفهم ذلك انظر إلى الأمر التالى :

LD B,(IX+5)

هذا الأمر سيقوم بتحميل المسجل B بمحتويات بايت الذاكرة التى عنوانها عبارة عن حاصل جمع محتويات المسجل IX زائد خمسة . لكى نوضح هذا الأمر افترض الوضع التالى :

B	IX	E105
05	E100	66

بعد تنفيذ الأمر LD B,(IX+5) سيصبح الوضع كالتالى :

B	IX	E105
66	E100	66

الرقم الذى يضاف على محتويات المسجل IX أو المسجل IY يسمى "الإزاحة" displacement وهذه الإزاحة تشغل بايت كاملة ولذلك فإن قيمتها ستتراوح بين +127 و -128 حيث الإزاحة الموجبة معناها إضافة إلى محتويات مسجل الفهرسة (IX, IY) وأما الإزاحة السالبة فمعناها طرح من مسجل الفهرسة .

الشفرات الست عشرية لجميع هذه الأوامر موجودة في جداول الأوامر الموضحة في نهاية هذا الفصل .

3-5 تمارين

استخدم قائمة أوامر الانتقال الخاصة بالشريحة Z80 لحل التمارين الموجودة في الجزء 3-4 في الفصل السابق .

4-5 مجموعة أوامر الحساب

Arithmetic Instructions

سندرس في هذا الجزء بعض الأوامر التي تقوم بإجراء العمليات الحسابية الأولية وهي الجمع والطرح ، وكما علمنا من قبل فإن مسجل التراكم لا بد وأن يكون طرفا في أى عملية من هذه العمليات كما أن نتيجة هذه العملية سواء كانت جمعا أو طرحا تكون دائما موجودة في مسجل التراكم A . هناك أيضا خاصية مهمة في مجموعة أوامر الحساب (ومثلها أيضا أوامر المنطق كما سنرى) وهي أنه نتيجة تنفيذ أى أمر من هذه الأوامر فإن الأعلام الموجودة في مسجل الحالة Status Register تتأثر بهذه النتيجة . راجع مسجل الحالة ومحتوياته ومتى يكون أى علم من أعلامه يساوى صفرا أو واحدا وذلك في الفصل الثانى . كما ذكرنا فإن طرفا من طرفى العملية الحسابية أو معامل من معاملها لا بد وأن يكون موضوعا في المسجل A وأما الطرف الثانى أو المعامل الثانى فإن مصدره سيكون واحدا من أربعة أماكن موضحة كالتالى :

<u>مصدر المعامل الثانى للعملية الحسابية</u>	<u>الرمز المستخدم</u>
مسجل 8 بت من مسجلات المعالج .	reg
بايت من بايتات الذاكرة عنوانها فى HL .	(HL)
ثابت أو معلومة فورية مكونة من 8 بت .	data8
بايت من بايتات الذاكرة معنونة بطريقة الفهرسة باستخدام المسجلين IX أو IY	(IX+d) (IY+d)

5-4-1 الأوامر ADD و SUB

الصورة العامة لأمر الجمع ADD هي :

ADD A,reg

$A \leftarrow A + \text{reg}$

حيث سيقوم هذا الأمر بجمع محتويات المسجل reg مع محتويات المسجل A ووضع النتيجة في المسجل A مع التأثير على الأعلام . لاحظ أننا في حالة أسمبلر الشريحة 8085 كنا نكتب هذا الأمر ADD reg بدون ذكر المسجل A على أساس أنه بديهى أن المعامل الثانى للعملية يكون فى المسجل A ، أما فى أسمبلر الشريحة Z80 فلا بد من ذكر طرفى العملية الحسابية بالرغم من أن أحدهما يكون دائما المسجل A وإن كانت بعض المراجع تهمل ذلك . الصور العامة الأخرى لأمر الجمع ستكون كالتالى :

ADD A,(HL)

حيث سيقوم هذا الأمر بجمع محتويات بايت الذاكرة التى يوجد عنوانها فى الزوج HL مع محتويات المسجل A وتوضع النتيجة فى المسجل A .

ADD A,data8

حيث سيجمع الثابت data8 مع المسجل A وتوضع النتيجة فى المسجل A .

ADD A,(IX+d)

ADD A,(IY+d)

حيث سيجمع محتويات المسجل A مع محتويات بايت الذاكرة التى عنوانها عبارة عن محتويات المسجل IX أو IY زائد الإزاحة d وتوضع النتيجة فى المسجل A . بنفس الطريقة يمكن كتابة الصورة العامة لأمر الطرح تبعا لمصدر المعامل الثانى كما يلى :

SUB A,reg

$A \leftarrow A - \text{reg}$

SUB A,(HL)

$A \leftarrow A - (\text{HL})$

SUB A,data8

$A \leftarrow A - \text{data8}$

SUB A,(IX+d)

SUB A,(IY+d)

$A \leftarrow A - (\text{IX}+d)$

$A \leftarrow A - (\text{IY}+d)$

فى جميع هذه الأوامر يتم طرح المعامل الثانى (reg) ، (HL) ، (data8) ، ((IX+d),(IY+d)) من المسجل A وتوضع النتيجة فى المسجل A ، أى أننا نؤكد هنا على أن المطروح منه يكون دائما المسجل A . لاحظ أن المسجل A يذكر مع هذا الأمر أيضا وذلك بالطبع لا دخل للمستخدم فيه ولكنه من الشروط التى

يفرضها الأسبيلر (كما في بعض المراجع) ، سنهمل ذكر الشفرات الست عشرية للأوامر ابتداء من هذا الموضع ومن يريد التعرف عليها أو استخدامها فعليه الاستعانة بالأشكال الخاصة بالأوامر في نهاية هذا الفصل .

مثال 5-2

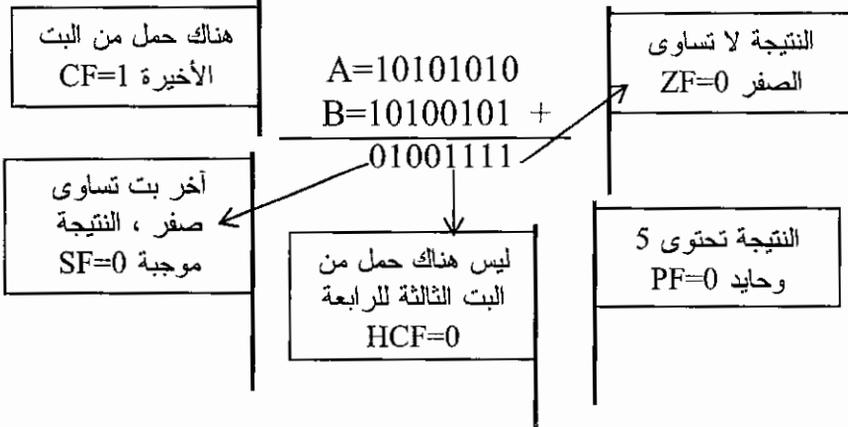
افتراض المحتويات الآتية للمسجلين A و B قبل تنفيذ الأمرين ADD و SUB :



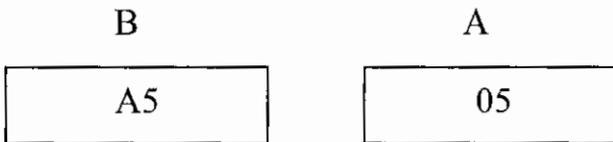
بعد تنفيذ الأمر ADD A,B ستصبح المحتويات كالتالي:



ولكى نرى كيفية تأثر الأعلام بنتيجة هذه العملية سنجرى عملية الجمع على الشفرات الثنائية لكل من الرقمين كما يلي :



الآن افترض أننا نفذنا أمر الطرح SUB B على المحتويات الأولى للمسجلين أى A=AAH و B=A5H فإنه بعد تنفيذ هذا الأمر ستصبح محتويات المسجلين كالتالي :



ولكى نرى كيف تمت عملية الطرح وكيف تأثرت الأعلام سنجرى عملية الطرح على الشفرات الثنائية لمحتويات المسجلين A و B كالتالى :

كما نعلم فإن عملية الطرح الثنائى يتم تحويلها إلى عملية جمع حيث سنجمع محتويات المسجل A (المطروح منه) مع المتمم الثنائى لمحتويات المسجل B (المطروح) (انظر الملحق الأول فى نهاية الكتاب لمراجعة عمليات الجمع والطرح الثنائى) . المتمم الثنائى لمحتويات المسجل B (10100101) هو 01011011 وبذلك تصبح عملية الطرح عملية جمع كالتالى :

$$A = 10101010$$

$$B + \text{المتمم الثنائى لمحتويات المسجل B} = 01011011 + 00000101$$

وبناء على ذلك ستكون الأعلام كالتالى :

- طالما أن البت الأولى لا تساوى صفرا فالنتيجة لا تساوى صفرا ويكون علم الصفر $ZF=0$.
- تحتوى النتيجة على عدد زوجى من الواحد (اثنين) لذلك سيكون علم الباريتى واحدا ، $PF=1$.
- هناك حمل من البت الثالثة إلى البت الرابعة فى حالة الجمع لذلك فعلم الحمل النصفى $HCF=0$.
- آخر بت (رقم 7) تساوى صفرا ، لذلك فالنتيجة موجبة وعلم الإشارة يكون دائما مساويا لمحتويات آخر بت ، إذن $SF=0$.
- المفروض فى عمليات الطرح يهمننا أن نعرف إذا كان هناك استلاف أم لا لأنه فى عملية الطرح لن يكون هناك حمل . بما أن عملية الطرح قد حولت إلى عملية جمع لذلك فإنه إذا كان هناك حمل فى عملية الجمع فان ذلك يعنى أنه لن يكون هناك استلاف فى عملية الطرح وسيكون علم الحمل $CYF=0$ وهى الحالة التى معنا الآن والعكس صحيح إذا لم يكن هناك حمل فى عملية الجمع ، وهذا هو ما طبقناه فى حالة العلم HCF .

2-4-5 الأمان ADC و SBC

بالنسبة للأمر ADC فإنه يجمع المعامل الثنائى سواء كان فى مسجل أو ذاكرة أو قيمة فورية مع محتويات المسجل A مع محتويات علم الحمل CY (صفر أو واحد) ويضع النتيجة فى المسجل A . الصورة العامة لهذه الأوامر وعلى حسب مصدر المعامل الثنائى ستكون كما يلى :

ADC A,reg

$$A \leftarrow A + CY + \text{reg}$$

ADC A,(HL)

$$A \leftarrow A + CY + (HL)$$

ADC A,data8

$A \leftarrow A + CY + data8$

ADC A,(IX+d)

$A \leftarrow A + CY + (IX+d)$

ADC A,(IY+d)

$A \leftarrow A + CY + (IY+d)$

يمكننا الآن تكرار نفس القول بالنسبة لأمر الطرح SBC حيث يقوم هذا الأمر بطرح المعامل الثاني سواء كان في مسجل أو ذاكرة أو قيمة فورية مع محتويات علم الحمل CY (صفر أو واحد) من المسجل A ثم توضع نتيجة الطرح في المسجل A ، نؤكد هنا على أن المطروح منه دائما هو المسجل A . الصورة العامة لهذه الأوامر على حسب مصدر المعامل الثاني ستكون كالتالى :

SBC A,reg

$A \leftarrow A - CY - reg$

SBC A,(HL)

$A \leftarrow A - CY - (HL)$

SBC A,data8

$A \leftarrow A - CY - data8$

SBC A,(IX+d)

$A \leftarrow A - CY - (IX+d)$

SBC A,(IY+d)

$A \leftarrow A - CY - (IY+d)$

مثال 3-5

المطلوب جمع الرقمين 23F9H و 9A35H ووضع نتيجة الجمع في أماكن الذاكرة E100 و E101 و E102 .

هذا المثال هو المثال رقم 4-11 وقد سبق حله كتطبيق على أمر الجمع مع الحمل في حالة الشريحة 8085 ويبين شكل (4-6) رسما توضيحيا ومخطط السير والبرنامج لطريقة حل هذا المثال ، لذلك يمكن مراجعته أولا وسنعيد كتابة البرنامج فقط بلغة الأسمبلى الخاصة بالشريحة Z80 في شكل (5-7) . قبل أن نتترك أوامر الجمع والطرح يجب أن نفهم جيدا متى يكون من الضروري استخدام الأمر ADC ؟ ومتى يكون من الضروري عدم استخدامه؟ مثلا في المثال السابق كان من الضروري عدم استخدام الأمر ADC في عملية الجمع الأولى (E + C) ولكن في هذه الحالة لا بد من استخدام الأمر ADD خوفا من أن يكون علم الحمل CY به واحد من أى عملية سابقة ونحن لا ندرى فيجمع مع عملية الجمع الأولى إذا استخدمنا الأمر ADC وتكون النتيجة خاطئة . أما فى

عملية الجمع الثانية (D + B + CY) فإنه لا بد من استخدام الأمر ADC لأننا نريد هنا أن نأخذ قيمة علم الحمل في الاعتبار .

E000	E001	LD C,F9H	
E002	E003	LD B,23H	
E004	E005	LD E,35H	
E006	E007	LD D,9AH	
	E008	LD A,C	
	E009	ADD A,E	
E00A	E00B	E00C	LD (E100),A
	E00D	LD A,B	
	E00E	ADC A,D	
E00F	E010	E011	LD (E101),A
	E012	E013	LD A,00
	E014	ADC A,A	
E015	E016	E017	LD (E102),A

شكل (5-7) برنامج المثال 3-5

3-4-5 الأوامر INC و DEC

هذان الأوامر يستخدمان لزيادة أو إنقاص واحد على أو من محتويات مسجل أو بايت من بايتات الذاكرة . الصورة العامة للأمر INC على حسب مكان المعلومة يمكن كتابتها كالتالي :

INC reg

$reg \leftarrow reg + 1$

INC (HL)

$(HL) \leftarrow (HL) + 1$

INC (IX+d)

$(IX+d) \leftarrow (IX+d) + 1$

INC (IY+d)

$(IY+d) \leftarrow (IY+d) + 1$

بنفس الطريقة يمكن كتابة الصورة العامة للأمر DEC كما يلي :

DEC reg

$reg \leftarrow reg - 1$

DEC (HL)

$(HL) \leftarrow (HL) - 1$

DEC (IX+d)

$(IX+d) \leftarrow (IX+d) - 1$

DEC (IY+d)
(IY+d) ← (IY+d) - 1

5-4-4 العمليات الحسابية على أزواج المسجلات

عند إجراء العمليات الحسابية على أزواج المسجلات يلعب الزوج HL دور مسجل التراكم من حيث أن المعامل الأول في العملية الحسابية لا بد وأن يكون في الزوج HL ونتيجة العملية الحسابية تذهب دائما إلى الزوج HL . يجب أن نعلم أنه عند إجراء العمليات الحسابية على أزواج المسجلات أن الأعلام لا تتأثر بهذه العمليات في الكثير من الأحيان ويجب النظر في حالة كل أمر منفصلة . الصورة العامة للأمرين INC و DEC في هذه الحالة هي :

INC rp
rp ← rp + 1
DEC rp
rp ← rp - 1

حيث rp ترمز لأي زوج من أزواج المسجلات SP, HL, DE, BC أو مسجل من المسجلات ال 16 بت وهي المسجل IX أو المسجل IY . كأمثلة على ذلك انظر إلى الأوامر التالية :

INC HL
DEC SP
INC IX

الأمر الأول سيزيد واحدا على محتويات المسجلين HL والثاني سينقص واحدا من محتويات المسجل SP والثالث سيزيد واحدا على محتويات المسجل IX . الأوامر INC rp و DEC rp ليس لها تأثير على الأعلام . من أوامر الجمع والطرح التي تجرى على أزواج المسجلات ما يلي :

ADD HL, rp
HL ← HL + rp
ADC HL, rp
HL ← HL + rp + CY
SBC HL, rp
HL ← HL - rp - CY

في جميع هذه الأوامر ترمز rp لأي زوج من الأزواج SP, HL, DE, BC ما عدا المسجلين IX, IY فلا يمكن استخدامهما مع هذه الأوامر . لاحظ أيضا أن أمر الطرح الوحيد المتاح هو أمر الطرح مع الحمل SBC ولذلك فإنه عند إجراء أى عملية طرح بدون أخذ علم الحمل في الحسبان يجب في هذه الحالة التأكد من أن علم الحمل يساوى صفرا . الأمر ADD HL, rp ليس له تأثير على الأعلام وأما الأمران ADC HL, rp و SBC HL, rp فيؤثران على الأعلام ما عدا علم

الحمل النصفى HC . نؤكد هنا على أن المطروح منه فى الأوامر السابقه هو المسجلين HL . هناك بعض الأوامر التى تسمح للمسجل IX أو المسجل IY بأن يلعب دور مسجل التراكم فى عمليات الجمع على أزواج المسجلات ، وهذه الأوامر هى :

IX ← IX + BC	ADD IX,BC
IX ← IX + DE	ADD IX,DE
IX ← IX + SP	ADD IX,SP
IX ← IX + IX	ADD IX,IX
IY ← IY + BC	ADD IY,BC
IY ← IY + DE	ADD IY,DE
IY ← IY + SP	ADD IY,SP
IY ← IY + IY	ADD IY,IY

جميع هذه الأوامر تؤثر على الأعلام ما عدا علم الحمل النصفى HC وأيضا جميع هذه الأوامر ليس لها نظير لعملية الطرح .

5-4-5 أمر المقارنة Compare Instruction

هناك أمر واحد فقط للمقارنة حيث أن عملية المقارنة لا تجرى على أى معلومة مكونة من 16 بت . لكى تتم عملية المقارنة فإن أحد المعاملين لابد وأن يكون فى المسجل A والمعامل الآخر يكون إما فى مسجل من مسجلات المعالج أو فى بايت من بايتات الذاكرة . عند تنفيذ أمر المقارنة يقوم المعالج بطرح محتويات المعامل الثانى من محتويات المسجل A وتهمل نتيجة الطرح تماما ولا تتغير محتويات المسجل A نتيجة هذه العملية ولكن الذى يتأثر فقط بهذه العملية هو الأعلام . الصورة العامة لأمر المقارنة وعلى حسب مصدر المعامل الثانى يمكن كتابتها كما يلى :

A - reg	CP reg
A - (HL)	CP (HL)
A - data8	CP data8
A - (IX+d)	CP (IX+d)
A - (IY+d)	CP (IY+d)

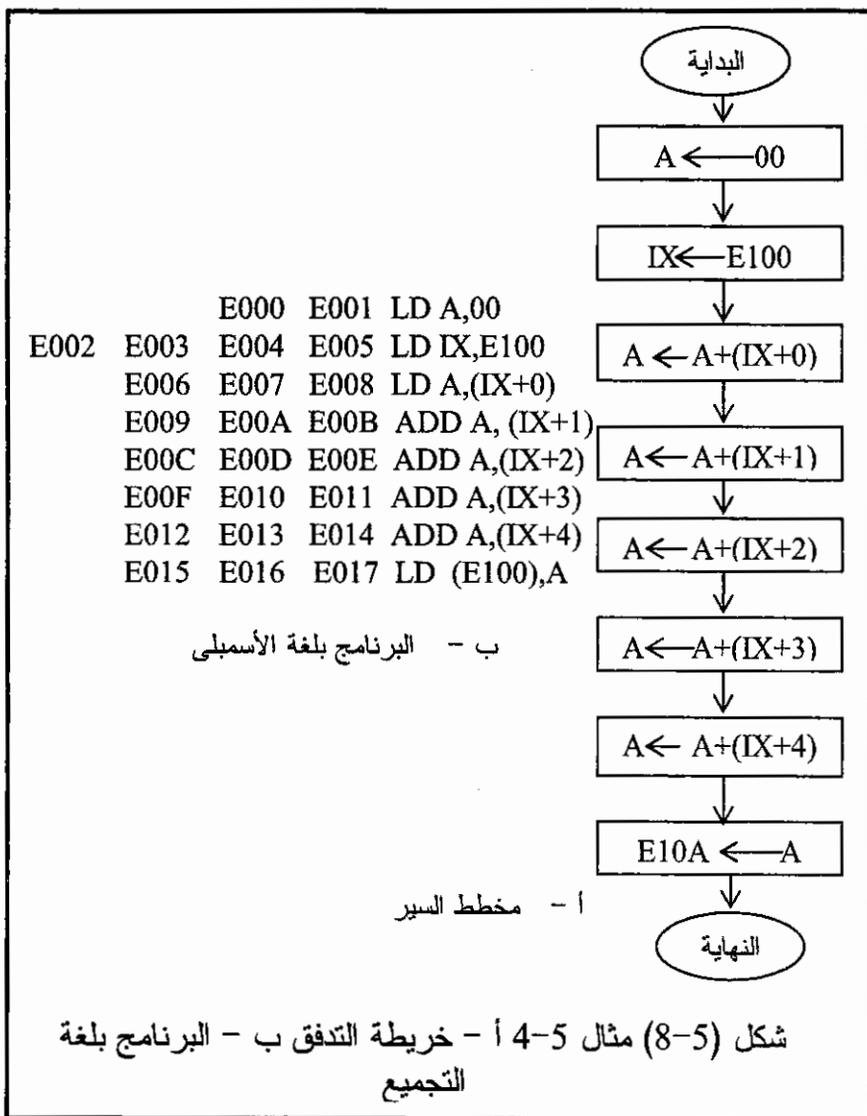
لاحظ أننا لم نكتب السهم الذى يوضح أين تذهب نتيجة عملية الطرح على أساس أن النتيجة تهمل كما ذكرنا . كامثلة على ذلك انظر إلى الأوامر التالية :

CP B
CP 3FH
CP (IY+9)

حيث سيقارن الأمر الأول محتويات المسجل B مع محتويات المسجل A وسيقارن الأمر الثانى محتويات المسجل A مع الثابت أو القيمة الفورية 3FH

وأما الأمر الثالث فسيقارن محتويات المسجل A مع محتويات بايت الذاكرة التي عنوانها تسعة زائد محتويات المسجل IX .

مثال 4-5



اكتب برنامجا يجمع محتويات الخمس بايتات E100, E101, E102, E103, E104, ويضع النتيجة في البايث E10A على اعتبار أن النتيجة لن تزيد عن بايت واحدة ، أى لن يكون هناك حمل على الإطلاق . هذا المثال من الممكن أن

يكون تدريباً جيداً على طرق الاتصال بالذاكرة التي تمت دراستها حتى الآن .
شكل (5-8) يبين مخطط السير والبرنامج لهذا المثال مستخدمين طريقة
الفهرسة مع المسجل IX للاتصال بالذاكرة ، حاول كتابة البرنامج مرة أخرى
مستخدماً الطريقة غير المباشرة مع الزوج HL .

5-5 تمارين

حل التمارين الموجودة في الجزء 4-5 في الفصل الرابع مستخدماً أوامر لغة
الأسمبلي الخاصة بالشريحة Z80 .

5-6 مجموعة أوامر القفز

Jump Instructions

القاعدة العامة أن المعالج يقوم بتنفيذ البرنامج حسب ترتيب الأوامر الموجودة
فيه من أول البرنامج إلى نهايته . ولقد كنا حريصين في جميع الأمثلة السابقة
على الحفاظ على هذه القاعدة ، ولكن هناك بعض المواقف أو قل بعض
التطبيقات التي تتطلب الخروج على هذه القاعدة كأن يطلب منك مثلاً تنفيذ
عملية معينة (عدد من الأوامر) عدد معين أو حتى عدد لا نهائي من المرات .
فعندما يكون المعالج مثلاً مراقباً لدرجة الحرارة في عملية صناعية معينة فإن
عليه أن يقرأ درجة الحرارة ويقارنها بدرجة حرارة مخزنة في الذاكرة كمرجع
وإذا زادت الحرارة عن حد معين يقوم المعالج بضرب جرس إنذار ، وإذا
نقصت عن حد معين يشغل سخان لزيادتها ، مثل هذا البرنامج سيكون عبارة
عن مجموعة من الأوامر التي تنفذ إلى ما لانهاية طالما أن المعالج يراقب درجة
الحرارة . لقد أتاح المعالج هذه العملية بتوفير بعض الأوامر التي تمكنك
كمبرمج من القفز بعملية التنفيذ من مكان لآخر خلال البرنامج ، وهناك ثلاثة
أنواع من القفز توفرها الشريحة Z80 وهي كما يلي :

5-6-1 القفز غير المشروط Unconditional jump

عند تنفيذ أي عملية قفز غير مشروط ينتقل المعالج بعملية التنفيذ إلى المكان
الجديد دون أي قيد أو شرط ، وهناك أمر واحد فقط من أوامر الشريحة Z80
يقوم بهذه العملية والصورة العامة لهذا الأمر هي :

JP addr

عند تنفيذ هذا الأمر يوضع العنوان addr الذي سيتم القفز إليه في عداد البرنامج
PC فيصبح الأمر الموجود عند هذا العنوان هو الأمر الذي عليه الدور في

التنفيذ . لاحظ أن هذا الأمر يتكون من ثلاث بايتات واحدة هي شفرة الأمر واثنتان للعنوان addr الذى سيتم القفز إليه . إن القفز باستخدام الأمر JP addr قد يكون إلى الأمام فى البرنامج وقد يكون إلى الخلف . إذا كان القفز إلى الأمام سينتج عن ذلك وجود جزء من البرنامج لن ينفذ على الإطلاق وهو الجزء الذى يقع بين أمر القفز JP addr والأمر الذى سيتم القفز إليه . أما إذا كان القفز إلى الخلف فإنه سينتج عن ذلك ما يسمى بالحلقة اللانهائية Infinite loop والتي سيستمر المعالج فى تنفيذها إلى ما لانهاية . شكل (4-8) يبين خريطة تدفق لعملية القفز غير المشروط بنوعها الأمامى والخلفى .

5-6-2 القفز المشروط Conditional jump

كما يوحي الاسم فإنه فى هذا النوع من القفز لن يتم القفز إلا إذا تحقق شرط معين ، أما إذا لم يتحقق هذا الشرط فإن البرنامج يتم تنفيذه فى التتابع الطبيعى حيث سينفذ الأمر الذى بعد أمر القفز مباشرة . إن شروط القفز توضع دائما على الأعلام التى فى مسجل الحالة SR ، فيمكنك مثلا أن تجعل القفز مشروطا بأن تكون النتيجة صفرا أو تجعله مشروطا بأن تكون النتيجة سالبة وهكذا . حيث أن هناك خمسة أعلام واحد منها وهو علم الحمل النصفى HC لا يستخدم كشرط فى عمليات القفز فإنه يتبقى أربعة أعلام يمكن أن تستخدم فى أوامر القفز المشروط كما يلي :

ZF=1	JPZ addr	• افقرز إذا كانت النتيجة صفرا
ZF=0	JPNZ addr	• افقرز إذا كانت النتيجة ليست صفرا
SF=1	JPM addr	• افقرز إذا كانت النتيجة سالبة
SF=0	JPP addr	• افقرز إذا كانت النتيجة موجبة
CF=1	JPC addr	• افقرز إذا كان هناك حمل
CF=0	JPNC addr	• افقرز إذا لم يكن هناك حمل
PF=1	JPPE addr	• افقرز إذا كانت الباريتى زوجية
PF=0	JPPO addr	• افقرز إذا كانت الباريتى فردية

لاحظ أن عدد هذه الأوامر ثمانية ، إثنان منها لكل علم من الأعلام الأربعة تمثل جميع الحالات التى يمكن أن يكون فيها هذا العلم صفرا أو واحدا . أيضا جميع هذه الأوامر لا بد وأن تتكون من ثلاث بايتات واحدة هي شفرة الأمر op code واثنتان للعنوان الذى سيتم القفز إليه . إن النتيجة التى سيتوقف عليها أمر القفز هى آخر نتيجة تأثرت بها الأعلام ، ولذلك فإنه قبل أن نكتب أى أمر من أوامر القفز المشروط يجب أن ندرس جيدا هل الأمر السابق لأمر القفز يؤثر على الأعلام أم لا .

5-6-3 القفز النسبي Relative jump

هناك أنواع أخرى من القفز متاحة لدى المعالج Z80 مثل القفز النسبي والقفز للبرامج الفرعية والعودة منها وسنترك الكلام عن هذه الأنواع حيث سيتم شرحها بالتفصيل في فصول خاصة بذلك وبالذات القفز للبرامج الفرعية .

مثال 5-5

اكتب برنامجا يقرأ محتويات البايث E100 باستمرار إلى ما لانهاية ثم يختبر هذه المحتويات بحيث إذا كانت صفرا يضع واحدا في المسجل B وإذا كانت سالبة يضع اثنين في المسجل B وإذا كانت موجبة يضع أربعة في نفس المسجل . شكل (4-9) في الفصل السابق يبين مخطط السير لهذا البرنامج وسنعيد فقط كتابة البرنامج بلغة الأسمبلى الخاصة بالشريحة Z80 كما في شكل (5-9) .

E000	E001	E002	LD HL,E100
	E003	E004	LD A,00
	E005	ADD	A,(HL)
E006	E007	E008	JP NZ,E00E
	E009	E00A	LD B,01
E00B	E00C	E00D	JP E000
E00E	E00F	E010	JP P,E016
	E011	E012	LD B,02
E013	E014	E015	JP E000
	E016	E017	LD B,04
E018	E019	E01A	JP E000

شكل (5-9) برنامج المثال 5-5

5-7 مهمة أخرى للأسمبلى

المهمة الوحيدة التي عرفناها للأسمبلى حتى الآن هي مهمة تحويل شفرات الأسمبلى إلى شفرات ثنائية أو لغة ماكينة ، ولكن لحسن الحظ فإن هناك مهام أخرى يستطيع الأسمبلى القيام بها ومن شأن هذه المهام أن تريح المبرمج وتوفر عليه الكثير من المجهود . جزء 4-7 في الفصل السابق تناول هذه المهام كما

تناول أيضا عملية تقسيم أى أمر من أوامر لغة الأسمبلى إلى أجزاء مختلفة وكيف يتعرف الأسمبلر على هذه الأجزاء ، لذلك فإننا لن نكرر هذا الجزء هنا ولكن نحيل القارئ لمراجعته فى الفصل السابق مع الأخذ فى الاعتبار الفوارق البسيطة بين شفرات الأسمبلى الخاصة بالشريحة Z80 والشريحة intel8085 .

8-5 أوامر الإدخال والإخراج Input Output Instructions

إلى الآن رأينا كيف نبرمج شريحة المعالج وكيف نحرك المعلومات داخلها من مسجل إلى مسجل آخر ومن أى مسجل إلى الذاكرة والعكس ، ولكن لم نعرف حتى الآن كيف نظهر معلومة معينة على شاشة عرض مثلا أيا كان نوع هذه الشاشة ، أو كيف ندخل معلومة إلى المعالج من خلال لوحة مفاتيح على سبيل المثال . إن لوحة المفاتيح وشاشة العرض يعتبران مثالين من العديد من الأمثلة التى تحتاج إلى عمليات الإخراج والإدخال . حينما يستخدم المعالج للتحكم فى أى متغير فى عملية صناعية وليكن مثلا درجة الحرارة فإنه لابد من إدخال درجة الحرارة إلى المعالج بعد تهيئتها ووضعها فى الصورة المناسبة لذلك ، وكذلك إذا أراد المعالج رفع درجة حرارة العملية الصناعية أو ضرب جرس إنذار فإنه يخرج إشارة معينة على بوابة إخراج تؤخذ وتهدأ فى الصورة المناسبة للجهاز الذى ستذهب إليه سواء كان سخانا أو جرسا . إن جميع عمليات الإدخال والإخراج تتم من خلال ما يسمى ببوابات الإدخال والإخراج والتعامل مع هذه البوابات دائما ينقسم إلى قسمين : قسم خاص بالبناء الإلكتروني لهذه البوابات وكيفية توصيلها مع المعالج وهذا القسم سندرسه بالتفصيل فى فصل قادم إن شاء الله ، والقسم الآخر هو كيفية برمجة المعالج للتعامل مع هذه البوابات وهو موضوع دراستنا فى هذا الجزء حيث سندرس الأوامر الخاصة بذلك وسنفترض فى دراستنا لهذا الجزء أن القارئ لديه على الأقل بوابة إدخال input port وبوابة إخراج output port موصلين فى الميكروكومبيوتر الذى يستخدمه فى عملية التدريب وكتابة البرامج .

1-8-5 أوامر الإدخال Input Instructions

الصورة العامة لأمر الإدخال هي :

IN A, no

محتويات البوابة رقم no. ← المسجل A

حيث IN هي اختصار لكلمة Input بمعنى ادخل ، وسيقوم هذا الأمر بإدخال المعلومة الموجودة على بوابة الإدخال رقم no. إلى مسجل التراكم A . لاحظ

أن عملية الإدخال بهذا الأمر تكون دائما على المسجل A حيث يمكن نقل المعلومة بعد ذلك إلى أى مكان آخر . هذا الأمر يتكون من اثنين بايت ، واحدة هي شفرة الأمر op code والأخرى هي رقم البوابة التى سيتم التعامل معها . ولذلك فإنه طالما أن رقم البوابة يشغل بايت كاملة فإن ذلك يعنى أنه يمكن التعامل مع $2^8 = 256$ بوابة إدخال تبدأ من البوابة رقم 00H إلى البوابة رقم FFH . انظر شفرة هذا الأمر فى جداول الأوامر فى نهاية هذا الفصل . هناك طريقة غير مباشرة للتعامل مع بوابات الإدخال والصورة العامة لها كالتالى :

IN reg,(C)

بوابة الأذخال التى رقمها فى المسجل C ← المسجل reg

حيث سيقوم هذا الأمر بإدخال المعلومة الموجودة فى بوابة الإدخال التى رقمها فى المسجل C إلى المسجل reg الذى هو أى مسجل من مسجلات المعالج وهذه ميزة عظيمة لم تكن موجودة فى المعالج 8085 .

5-8-2 أوامر الأخراج Output Instruction

الصورة العامة لأمر الأخراج هى :

OUT no ,A

المسجل A ← البوابة رقم no.

حيث OUT هى اختصار لكلمة Output التى تعنى إخراج ، وسيقوم هذا الأمر بإخراج المعلومة الموجودة فى المسجل A إلى بوابة الإخراج التى رقمها no . هذا الأمر أيضا يشغل اثنين بايت واحدة هي شفرة الأمر والأخرى هي رقم البوابة المراد التعامل معها ، ولذلك فإنه بهذا الأمر يمكن التعامل مع $2^8 = 256$ بوابة إخراج تبدأ من البوابة رقم 00H وتنتهى بالبوابة رقم FFH . الطريقة الغير مباشرة لهذا الأمر هى :

OUT (C),reg

محتويات المسجل reg ← بوابة الإخراج التى رقمها فى المسجل C

حيث سيقوم هذا الأمر بإخراج المعلومة الموجودة فى المسجل reg الذى يمثل أى مسجل من مسجلات المعالج إلى بوابة الإخراج التى يوجد رقمها فى المسجل C.

مثال 5-6

إفترض أن لدينا خط إنتاج فى أحد المصانع تعبر عليه المنتجات ، وفى أثناء العبور فإن كل منتج يقطع خلية ضوئية فتعطى نبضة كهربية على خرجها . خرج هذه الخلية موصل على البت رقم 0 فى بوابة الإدخال رقم 00H والمطلوب هو كتابة برنامج يعد هذه المنتجات ويخرج العدد على بوابة الإخراج

رقم 00H . شكلى (4-11 و 4-12) فى الفصل السابق يبينان رسما توضيحيا ومخطط السير لهذا المثال ، أما البرنامج فتمت إعادته كما فى شكل (5-10) .

5-9 مجموعة أوامر المنطق

Logic Instructions

العمليات المنطقية التى يستطيع المعالج Z80 القيام بها هى العمليات XOR, NOT, OR, AND وسنكتفى هنا بعرض الصورة العامة لهذه الأوامر على أن يقوم القارئ بمراجعتها فى جداول الأوامر الملحقة فى آخر الفصل . كما ذكرنا سابقا فإن العمليات المنطقية مثلها مثل العمليات الحسابية لا بد وأن يكون المسجل A طرفا فيها كما أن النتيجة توضع فى المسجل A .

```
LD B,00; المسجل B سيكون عداد للمنتج
LD C,00; المسجل C يحتوى رقم البوابة التى سنخرج عليها
HERE1: IN A,00 ; قراءة بوابة الإدخال إلى المسجل A
CP 00 ; مقارنة الإشارة بصفر
JP Z,HERE1; طالما أن الإشارة صفر يستمر فى هذه الحلقة
INC B; عند اختلاف الإشارة عن الصفر يزيد B بواحد
HERE2: IN A,00; حلقة انتظار إلى أن ترجع الإشارة للصفر
CP 01
JP Z,HERE2
OUT (C),B ; يخرج محتويات المسجل B على بوابة الإخراج 00
JP HERE1; يذهب للبداية ليقرا نبضة جديدة
```

شكل (5-10) برنامج المثال 5-6

جميع العمليات المنطقية تؤثر على الأعلام ما عدا علمي الحمل والحمل النصفى حيث يكونان دائما صفرا بعد أى عملية منطقية لأن الحمل والحمل النصفى غير معرف مع العمليات المنطقية .

الصورة العامة لأوامر العملية AND هى :

AND reg ←

المسجل A AND المسجل reg
 AND (HL)
 المسجل A ← المسجل AND (HL)
 AND data8
 المسجل A ← المسجل AND data8
 AND (IX+d)
 المسجل A ← المسجل AND (IX+d)
 AND (IY+d)
 المسجل A ← المسجل AND (IY+d)

بنفس الطريقة يمكن كتابة الصورة العامة لأوامر OR و XOR كما يلي :

OR reg
 OR (HL)
 OR data8
 OR (IX+d)
 OR (IY+d)
 XOR reg
 XOR (HL)
 XOR data8
 XOR (IX+d)
 XOR (IY+d)

هناك عملية NOT وهي لا تجرى إلا على المسجل A حيث يقلب كل صفر إلى واحد وكل واحد إلى صفر ، والصورة العامة لهذا الأمر هي :

CPL
 المسجل A ← عكس المسجل A
 هذه العملية تسمى عملية المتمم الأحادي وهناك عملية المتمم الثنائي المعرفة كالتالي :

المتمم الثنائي = 1 + المتمم الأحادي

وهناك أمر يقوم بعملية المتمم الثنائي وصورته العامة هي :

NEG
 المسجل A ← المتمم الثنائي للمسجل A
 إن للمتمم الثنائي أهمية خاصة في تحويل عمليات الطرح إلى جمع كما هو مشروح بالتفصيل في الملحق رقم 1 في نهاية الكتاب .
 إلى هنا نكون قد انتهينا من العرض التفصيلي لمعظم أوامر الشريحة Z80 على أننا سنعرض هذه الأوامر أولاً في صورة مجموعات كما في الأشكال (5-11 إلى 5-20) ثم سنعرض الأوامر مرتبة ترتيباً أبجدياً كما في شكل (5-21) .

LD	A,	B,	C,	D,	E,	H,	L,	(HL),	(IX+d),	(IY+d),
A	7F	47	4F	57	5F	67	6F	77	DD77dd	FD77dd
B	78	40	48	50	58	60	68	70	DD70dd	FD70dd
C	79	41	49	51	59	61	69	71	DD71dd	FD71dd
D	7A	42	4A	52	5A	62	6A	72	DD72dd	FD72dd
E	7B	43	4B	53	5B	63	6B	73	DD73dd	FD73dd
H	7C	44	4C	54	5C	64	6C	74	DD74dd	FD74dd
L	7D	45	4D	55	5D	65	6D	75	DD75dd	FD75dd
data8	3E xx	06 xx	0E xx	16 xx	1E xx	26 xx	2E xx	36xx	DD36dd xx	FD36ddxx
(HL)	7E	46	4E	56	5E	66	6E			
(IX+d)	7E	46	4E	56	5E	66	6E			*
(IY+d)	7E	46	4E	56	5E	66	6E			**

* جميع شفرات أوامر هذا الصف تسبقها DD ويعقبها dd مثلها مثل أوامر العمود (IX+d).

** جميع شفرات أوامر هذا الصف تسبقها FD ويعقبها dd مثلها مثل أوامر العمود (IY+d).

xx يقصد بها البايث الثانية من الأمر وهي data8

LD	BC,	DE,	HL,	SP,	IX,	IY,
(addr)	ED4B adr	ED5B adr	2A adr	ED7B adr	DD2A adr	FD2A adr
data16	01 dat16	11 dat16	21 dat16	31 dat16	DD21 dat16	FD21 dat16

LD (addr),	BC	DE	HL	SP	IX	IY
	ED43 adr	ED53 adr	22 adr	ED73 adr	DD22 adr	FD22 adr

	SR	BC	DE	HL	IX	IY
PUSH	F5	C5	D5	E5	DDE5	FDE5
POP	F1	C1	D1	E1	DDE1	FDE1

LD SP,	HL	IX	IY
	F9	DDF9	FDF9

شكل (5-11) مجموعة أوامر الانتقال للمعالج Z80

EX DE,HL	EB
EX (sp),HL	E3
EX (SP),IX	DDE3
EX (SP),IY	FDE3
EX SR,SR1	08
EXX	D9

شكل (5-12) مجموعة أوامر الاستبدال

	ADD A,	ADC A,	SUB	SBC A,	INC	DEC	CP
A	87	8F	97	9F	3C	3D	BF
B	80	88	90	98	04	05	B8
C	81	89	91	99	0C	0D	B9
D	82	8A	92	9A	14	15	BA
E	83	8B	93	9B	1C	1D	BB
H	84	8C	94	9C	24	25	BC
L	85	8D	95	9D	2C	2D	BD
(HL)	86	8E	96	9E	34	35	BE
data8	C6	CE	D6	DE			FE
	xx	xx	xx	xx			xx
(IX+d)	DD	DD	DD	DD	DD	DD	DD
	86	8E	96	9E	34	35	BE
	dd	dd	dd	dd	dd	dd	dd
(IY+d)	FD	FD	FD	FD	FD	FD	FD
	86	8E	96	9E	34	35	BE
	dd	dd	dd	dd	dd	dd	dd

شكل (5-13) مجموعة أوامر الحساب

	AND	OR	XOR
A	A7	B7	AF
B	A0	B0	A8
C	A1	B1	A9
D	A2	B2	AA
E	A3	B3	AB
H	A4	B4	AC
L	A5	B5	AD
(HL)	A6	B6	AE
data8	E6xx	F6xx	EExx
(IX+d)	DDA6dd	DDB6dd	DDAEdd
(IY+d)	FDA6dd	FDB6dd	FDAEdd

	CPL	NEG	CCF	SCF
A	2F	ED44	3F	37

شكل (5-14) مجموعة أوامر المنطق

	ADD HL,	ADC HL,	SBC HL,	ADD IX,	ADD IY,	INC	DEC
BC	09	ED 4A	ED 42	DD 09	FD 09	03	0B
DE	19	ED 5A	ED 52	DD 19	FD 19	13	1B
HL	29	ED 6A	ED 62			23	2B
SP	39	ED 7A	ED 72	DD39	FD 39	33	3B
IX				DD 29		DD 23	DD 2B
IY					FD 29	FD 23	FD 2B

شكل (5-15) أوامر حسابية على أزواج مسجلات

JP	JP Z	JP NZ	JP C	JP NC	JP M	JP P	JP PE	JP PO
C3	CA	C2	DA	D2	FA	F2	EA	E2
xx	xx	xx	xx	xx	xx	xx	xx	xx
xx	xx	xx	xx	xx	xx	xx	xx	xx

JR	JR Z	JR NZ	JR C	JR NC
18xx	28xx	20xx	38xx	30xx

CALL addr	CDxxxx
CALL Z,addr	CCxxxx
CALL NZ,addr	C4xxxx
CALL C,addr	DCxxxx
CALL NC,addr	D4xxxx
CALL M,addr	FCxxxx
CALL P,addr	F4xxxx
CALL PE,addr	ECxxxx
CALL PO,addr	E4xxxx

RET	C9
RET Z	C8
RET NZ	C0
RET C	D8
RET NC	D0
RET M	F8
RET P	F0
RET PE	E8
RET PO	E0

xxxx تمثل اثنين بايت للعنوان الذي سيتم القفز إليه
xx تمثل بايت واحدة للعنوان الذي سيتم القفز إليه في حالة القفز النسبي

شكل (5-16) لجلوعب أوالد القفد

الأمر BIT b,sss يجعل علم الصفر يساوي عكس البت رقم b في المسجل أو الذاكرة sss

BIT	0,	1,	2,	3,	4,	5,	6,	7,
A	CB 47	CB 4F	CB 57	CB 5F	CB 67	CB 6F	CB 77	CB 7F
B	CB 40	CB 48	CB 50	CB 58	CB 60	CB 68	CB 70	CB 78
C	CB 41	CB 49	CB 51	CB 59	CB 61	CB 69	CB 71	CB 79
D	CB 42	CB 4A	CB 52	CB 5A	CB 62	CB 6A	CB 72	CB 7A
E	CB 43	CB 4B	CB 53	CB 5B	CB 63	CB 6B	CB 73	CB 7B
H	CB 44	CB 4C	CB 54	CB 5C	CB 64	CB 6C	CB 74	CB 7C
L	CB 45	CB 4D	CB 55	CB 5D	CB 65	CB 6D	CB 75	CB 7D
(HL)	CB 46	CB 4E	CB 56	CB 5E	CB 66	CB 6E	CB 76	CB 7E
(IX+d)	DD CB d46	DD CB d4E	DD CB d56	DD CB d5E	DD CB d66	DD CB d6E	DD CB d76	DD CB d7E
(IY+d)	FD CB d46	FD CB d4E	FD CB d56	FD CB d5E	FD CB d66	FD CB d6E	FD CB d76	FD CB d7E

شكل (5-17) مجموعة اختبار و SET و RESET بت من بتات مسجل أو مكان في الذاكرة

الأمر SET b,sss يجعل البت رقم b في المسجل أو الذاكرة sss تساوى واحد

SET	0,	1,	2,	3,	4,	5,	6,	7,
A	CB C7	CB CF	CB D7	CB DF	CB E7	CB EF	CB F7	CB FF
B	CB C0	CB C8	CB D0	CB D8	CB E0	CB E8	CB F0	CB F8
C	CB C1	CB C9	CB D1	CB D9	CB E1	CB E9	CB F1	CB F9
D	CB C2	CB CA	CB D2	CB DA	CB E2	CB EA	CB F2	CB FA
E	CB C3	CB CB	CB D3	CB DB	CB E3	CB EB	CB F3	CB FB
H	CB C4	CB CC	CB D4	CB DC	CB E4	CB EC	CB F4	CB FC
L	CB C5	CB CD	CB D5	CB DD	CB E5	CB ED	CB F5	CB FD
(HL)	CB C6	CB CE	CB D6	CB DE	CB E6	CB EE	CB F6	CB FE
(IX+d)	DD CB dC6	DD CB dCE	DD CB dD6	DD CB dDE	DD CB dE6	DD CB dEE	DD CB dF6	DD CB dF E
(IY+d)	FD CB dC6	FD CB dCE	FD CB dD6	FD CB dDE	FD CB dE6	FD CB dEE	FD CB dF6	FD CB dF E

تابع شكل (5-17) مجموعة اختبار و SET و RESET بت من بتات مسجل أو مكان في الذاكرة

الأمر RES b,sss يجعل البت رقم b في المسجل أو الذاكرة sss تساوى صفر

RES	0,	1,	2,	3,	4,	5,	6,	7,
A	CB 87	CB 8F	CB 97	CB 9F	CB A7	CB AF	CB B7	CB BF
B	CB 80	CB 88	CB 90	CB 98	CB A0	CB A8	CB B0	CB B8
C	CB 81	CB 89	CB 91	CB 99	CB A1	CB A9	CB B1	CB B9
D	CB 82	CB 8A	CB 92	CB 9A	CB A2	CB AA	CB B2	CB BA
E	CB 83	CB 8B	CB 93	CB 9B	CB A3	CB AB	CB B3	CB BB
H	CB 84	CB 8C	CB 94	CB 9C	CB A4	CB AC	CB B4	CB BC
L	CB 85	CB 8D	CB 95	CB 9D	CB A5	CB AD	CB B5	CB BD
(HL)	CB 86	CB 8E	CB 96	CB 9E	CB A6	CB AE	CB B6	CB BE
(IX+d)	DD CB d86	DD CB d8E	DD CB d96	DD CB d9E	DD CB dA6	DD CB dA E	DD CB dB6	DD CB dBE
(IY+d)	FD CB d86	FD CB d8E	FD CB d96	FD CB d9E	FD CB dA6	FD CB dA E	FD CB dB6	FD CB dBE

تابع شكل (5-17) مجموعة اختبار و SET و RESET بت من بتات مسجل أو مكان في الذاكرة

	RL C	RRC	RL	RR	SLA	SRA	SRL
A	CB 07	CB 0F	CB 17	CB 1F	CB 27	CB 2F	CB 3F
B	CB 00	CB 08	CB 10	CB 18	CB 20	CB 28	CB 38
C	CB 01	CB 09	CB 11	CB 19	CB 21	CB 29	CB 39
D	CB 02	CB 0A	CB 12	CB 1A	CB 22	CB 2A	CB 3A
E	CB 03	CB 0B	CB 13	CB 1B	CB 23	CB 2B	CB 3B
H	CB 04	CB 0C	CB 14	CB 1C	CB 24	CB 2C	CB 3C
L	CB 05	CB 0D	CB 15	CB 1D	CB 25	CB 2D	CB 3D
(HL)	CB 06	CB 0E	CB 16	CB 1E	CB 26	CB 2E	CB 3E
(IX+d)	DD CB d06	DD CB d0E	DD CB d16	DD CB d1E	DD CB d26	DD CB d2E	DD CB d3E
(IY+d)	FD CB d06	FD CB d0E	FD CB d16	FD CB d1E	FD CB d26	FD CB d2E	FD CB d3E

أوامر خاصة بإزاحة أو دوران المسجل A فقط

RRA	RLA	RRCA	RLCA
1F	17	0F	07

شكل (5-18) مجموعة أوامر الأزاحة والدوران

IN	A,	B,	C,	D,	E,	H,	L,
Port No.	DBxx	----	----	----	----	----	----
(C)	ED78	ED40	ED48	ED50	ED58	ED60	ED68

OUT	Port No.,	(C),
A	D3xx	ED79
B	----	ED41
C	----	ED49
D	----	ED51
E	----	ED59
H	----	ED61
L	----	ED69

شكل (5-19) أوامر الإدخال والإخراج

No Operation, NOP لا تعمل شيء	00
HALT توقف	76
Disable Interrupt, DI إخماد المقاطعة	F3
Enable Interrupt, EI تنشيط المقاطعة	FB
IM0 تنشيط حالة المقاطعة رقم صفر	ED46
IM1 تنشيط حالة المقاطعة رقم صفر	ED56
IM2 تنشيط حالة المقاطعة رقم صفر	ED5E

شكل (5-20) مجموعة أوامر متفرقة

كانت هذه بعض أهم مجموعات الأوامر للمعالج Z80 والشائعة الاستخدام . شكل (5-21) يحتوي جميع أوامر الشريحة مرتبة ترتيباً أبجدياً مع نبذة عن ما يعمله كل أمر وعدد نبضات التزامن التي يأخذها (ن #) لكي يتم إحضاره من الذاكرة وتنفيذه والأعلام التي تتأثر بكل أمر وكذلك شفرة كل أمر حيث من هذه الشفرة يمكن استنتاج عدد بايتات الأمر . انظر الملاحظات الخاصة بشفرة الأوامر في نهاية هذا الشكل .

وظيفة الأمر	شفرة الأمر	# ن	الأعلام المتأثرة	شفرة الأسبيلي
$A \leftarrow A+CY+reg$	10001sss	4	ZSP CY HC	ADC A,reg
$A \leftarrow A+CY+(HL)$	8E	7	ZSP CY HC	ADC A,(HL)
$A \leftarrow A+CY+data8$	CE data8	7	ZSP CY HC	ADC A,data8
$A \leftarrow A+CY+(IY+d)$	FD 8E dd	19	ZSP CY HC	ADC A,(IY+d)
$A \leftarrow A+CY+(IX+d)$	DD 8E dd	19	ZSP CY HC	ADC A,(IX+d)
$HL \leftarrow HL+CY+rp$	ED 01rp1010	15	--- CY -	ADC HL,rp
$A \leftarrow A + Reg$	10000sss	4	ZSP CY HC	ADD A,reg
$A \leftarrow A + (HL)$	86	7	ZSP CY HC	ADD A,(HL)
$A \leftarrow A+data8$	C6 data8	7	ZSP CY HC	ADD A,data8
$A \leftarrow A+(IY+d)$	FD 88 dd	19	ZSP CY HC	ADD A,(IY+d)
$A \leftarrow A+(IX+d)$	DD 86 dd	19	ZSP CY HC	ADD A,(IX+d)
$HL \leftarrow HL+rp$	00rp1001	11	- - - CY -	ADD HL,rp
$IX \leftarrow IY + rp$	FD 00rp1001	15	- - - CY -	ADD IY,rp
$IX \leftarrow IX + rp$	DD 00rp1001	15	- - - CY -	ADD IX,rp
$A \leftarrow A \text{ AND } reg$	10100xxx	4	ZS P 0 0	AND reg
$A \leftarrow A \text{ AND } (HL)$	A6	7	ZS P 0 0	AND (HL)
$A \leftarrow A \text{ AND } data8$	E6 data8	7	ZS P 0 0	AND data8
$A \leftarrow A \text{ AND } (IY+d)$	FD A6 dd	19	ZS P 0 0	AND (IY+d)
$A \leftarrow A \text{ AND } (IX+d)$	DD A6 dd	19	ZS P 0 0	AND (IX+d)
عكس البت b في reg في ZF \leftarrow	CB 01bbbsss	9	Z - - - -	BIT b,reg
عكس البت b في (HL) في ZF \leftarrow	CB 01bbb110	12	Z - - - -	BIT b,(HL)
عكس البت b في (IY+D) في ZF \leftarrow	FD CB dd 01bbb110	20	Z - - - -	BIT b,(IY+d)
عكس البت b في (IX+D) في ZF \leftarrow	DD CB dd 01bbb110	20	Z - - - -	BIT b,(IX+d)
نداء غير مشروط لبرنامج فرعي	CD addr	17	-----	CALL addr
نداء مشروط بعلم الحمل=1	DC addr	10/17	-----	CALL C,addr
نداء مشروط بعلم إشارة=1	FC addr	10/17	-----	CALL M,addr
نداء مشروط بعلم الحمل=0	D4 addr	10/17	-----	CALL NC,addr
نداء مشروط بعلم الصفر=0	C4 addr	10/17	-----	CALL NZ,addr
نداء مشروط بعلم إشارة=0	F4 addr	10/17	-----	CALL P,addr
نداء مشروط بعلم باريتي=1	EC addr	10/17	-----	CALL PE,addr
نداء مشروط بعلم باريتي=0	E4 addr	10/17	-----	CALL PO,addr
نداء مشروط بعلم الصفر=1	CC addr	10/17	-----	CALL Z,addr
اعكس علم الحمل	3F	4	--- CY HC	CCF
مقارنة reg - A	10111sss	4	ZSP CY HC	CP reg
مقارنة (HL) - A	BE	7	ZSP CY HC	CP (HL)
مقارنة const - A	FE data8	7	ZSP CY HC	CP const
مقارنة (IY+d) - A	FD BE data8	19	ZSP CY HC	CP (IY+d)
مقارنة (IX+d) - A	DD BE data8	19	ZSP CY HC	CP (IX+d)
اعكس المسجل A	2F	4	-----	CPL

CPI	ZSP CY HC	16	ED A1	مقارنة A-(HL) BC ← BC-1 , HL ← HL+1
CPIR	ZSP CY HC	21/16	ED B1	كرر CPI إلى BC=0
CPD	ZSP CY HC	16	ED A9	مقارنة A-(HL) BC ← BC-1 , HL ← HL-1
CPDR	ZSP CY HC	21/16	ED B9	كرر CPD إلى BC=0
DAA	ZSP CY HC	4	27	حول المرمك للنظام العشري
DEC reg	ZSP -- HC	4	00ddd101	reg ← reg - 1
DEC (HL)	ZSP -- HC	7	35	(HL) ← (HL) - 1
DEC (IY+d)	ZSP -- HC	19	FD 35 data8	(IY+d) ← (IY+d) - 1
DEC (IX+d)	ZSP -- HC	19	DD 35 data8	(IX+d) ← (IX+d) - 1
DI	-----	4	F3	أهمل المقاطعة 0 ← IF
DEC rp	-----	6	00rp1011	rp ← rp - 1
DEC IY	-----	10	FD 2B	IY ← IY - 1
DEC IX	-----	10	DD 2B	IX ← IX - 1
DJNZ ddd	-----	8/13	10 ddd	قفز بمقدار ddd وإنفاص B بمقدار 1 إلى أن يصبح B=0
EI	-----	4	FB	اسمح بالمقاطعة 1 ← IF
EX DE,HL	-----	4	EB	HL ↔ DE
EX AF,AFI	-----	4	08	PSW1 ↔ PSW
EXX	-----	4	D9	BCDEHL1 ↔ BCDEHL
EX (SP),HL	-----	19	E3	(SP+1) → H (SP) → L
EX (SP),IX	-----	23	FD E3	(SP+1) → IY/H (SP) → IY/L
EX (SP),IX	-----	23	DD E3	(SP+1) → IX/H (SP) → IX/L
HALT		4	76	أوقف تنفيذ البرنامج
IM 0	-----	8	ED 46	حالة المقاطعة صفر
IM 1	-----	8	ED 56	حالة المقاطعة واحد
IM 2	-----	8	ED 5E	حالة المقاطعة اثنين
IN A,(no.)	-----	11	DB no8	البوابة رقم. no. ← A
IN reg,(C)	-----	12	ED 01ddd000	البوابة (C) ← reg
INI	-----	16	ED A2	port (C) → (HL) B-1 → B HL ← HL+1
INIR	-----	16/21	ED B2	كرر INI إلى أن B=0
IND	-----	16	ED AA	port(C) → (HL) B-1 → B HL ← HL-1
INDR	-----	16/21	ED BA	كرر IND إلى أن B=0
INC reg	Z S P - HC	4	00ddd100	reg ← reg + 1
INC (HL)	Z S P - HC	7	34	(HL) + 1 → (HL)
INC (IY+d)	Z S P - HC	19	FD 34 dd	(IY+d) ← (IY+d) + 1
INC (IX+d)	Z S P - HC	19	DD 34 dd	(IX+d) ← (IX+d) + 1
INC rp	-----	6	00xx0011	rp ← rp + 1

INC IY	-----	10	FD 23	IY ← IY + 1
INC IX	-----	10	DD 23	IX ← IX + 1
JP addr	-----	10	C3 addr	قفز غير مشروط
JP (HL)	-----	4	E9	قفز إلى عنوان في HL
JP (IX)	-----	8	DD E9	قفز إلى عنوان في IX
JP (IY)	-----	8	FD E9	قفز إلى عنوان في IY
JP Z,addr	-----	10	CA addr	قفز مشروط بعلم الصفر=1
JP NZ,addr	-----	10	C2 addr	قفز مشروط بعلم الصفر=0
JP C,addr	-----	10	DA addr	قفز مشروط بعلم الحمل=1
JP NC,addr	-----	10	D2 addr	قفز مشروط بعلم الحمل=0
JP PO,addr	-----	10	E2 addr	قفز مشروط بعلم باريتي=0
JP PE,addr	-----	10	EA addr	قفز مشروط بعلم باريتي=1
JP P,addr	-----	10	F2 addr	قفز مشروط بعلم اشارة=0
JP M,addr	-----	10	FA addr	قفز مشروط بعلم اشارة=1
JR dd	-----	10	18 dd	قفز نسبي غير مشروط
JR Z,dd	-----	7/12	28 dd	قفز نسبي مشروط Z=1
JR NZ,dd	-----	7/12	20 dd	قفز نسبي مشروط Z=0
JR C,dd	-----	7/12	38 dd	قفز نسبي مشروط CY=1
JR NC,dd	-----	7/12	30 dd	قفز نسبي مشروط CY=0
LD reg1,reg2	-----	4	01dddsss	reg2 → reg1
LD reg,(HL)	-----	7	01ddd110	(HL) → reg
LD (HL),reg	-----	7	01110sss	(HL) ← reg
LD reg,data8	-----	7	00ddd110 data8	reg ← data8
LD reg,(IY+d)	-----	19	FD 01ddd110 dd	reg ← (IY+d)
LD reg,(IX+d)	-----	19	DD 01ddd110 dd	reg ← (IX+d)
LD (IY+d),reg	-----	19	FD 01110sss dd	(IY+d) ← reg
LD (IX+d),reg	-----	19	DD 01110sss dd	(IX+d) ← reg
LD (HL),data8	-----	10	36 data8	(HL) ← data8
LD (IY+d),data8	-----	19	FD 36 dd data8	(IY+d) ← data8
LD (IX+d),data8	-----	19	DD 36 dd data8	(IX+d) ← data8
LD A,(addr)	-----	13	3A addr	A ← (addr)
LD (addr),A	-----	13	32 addr	(addr) ← A
LD (addr),BC	-----	20	ED 43 addr	addr ← C addr+1 ← B
LD (addr),DE	-----	20	ED 53 addr	addr ← E addr+1 ← D
LD (addr),HL	-----	20	22 addr	addr ← L addr+1 ← H
LD (addr),IX	-----	20	DD 22 addr	addr ← IX/L addr+1 ← IX/H
LD (addr),IY	-----	20	FD 22 addr	addr ← IY/L addr+1 ← IY/H
LD (addr),SP	-----	20	ED 73 addr	addr ← SP/L addr+1 ← SP/H
LD A,(BC)	-----	7	0A	A ← (BC)
LD A,(DE)	-----	7	1A	A ← (DE)
LD (BC),A	-----	7	02	A → (BC)
LD (DE),A	-----	7	12	A → (DE)
LD A,I	-----	9	ED 57	A ← I
LD I,A	-----	9	ED 47	I ← A
LD A,R	-----	9	ED 5F	A ← R
LD R,A	-----	9	ED 4F	R ← A

LD rp,data16	-----	10	00rp0001 data16	rp ← data16
LD IX,data16	-----	14	DD 21 data16	IX ← data16
LD rp,(addr)	-----	20	ED 01rp1011 addr	(addr+1) → B (addr) → C
LD HL,(addr)	-----	16	2A addr	(addr+1) → H (addr) → L
LD IX,(addr)	-----	20	DD 2A addr	(addr) → IX/L IX/H ← (addr+1)
LD IY,(addr)	-----	20	FD 2A addr	(addr) → IY/L IY/H ← (addr+1)
LD SP,HL	-----	16	F9	HL → SP
LD SP,IX	-----	10	DD F9	SP ← IX
LD SP,IY	-----	10	FD F9	SP ← IY
LDI	Z S P - -	16	ED A0	DE ← DE+1 (HL) → (DE) HL ← HL+1 BC ← BC-1
LDIR	Z S P - -	21/16	ED B0	نفس الأمر السابق إلى BC=0
LDD	Z S P - -	16	ED AB	DE ← DE-1 (HL) → (DE) BC ← BC-1 HL ← HL-1
LDDR	Z S P - -	21/16	ED BB	نفس الأمر السابق إلى BC=0
NEG	Z S P - -	8	ED 44	A ← A المتمم الثنائي ل
NOP	-----	4	00	لا تعمل شيئا No operation
OR reg	Z S P 0 0	4	10110sss	A ← A OR reg
OR (HL)	Z S P 0 0	7	B5	A ← A OR (HL)
OR data8	Z S P 0 0	7	F6 data8	A ← A OR data8
OR (IY+d)	Z S P 0 0	19	FD B6 dd	A ← A OR (IY+d)
OR (IX+d)	Z S P 0 0	19	DD B6 dd	A ← A OR (IX+d)
OUT (no.),A	-----	11	D3 no.8	port (no.) ← A
OUT (C),reg	-----	12	ED 01sss001	port (C) ← reg
OUTI	-----	16	ED A3	(HL) → port(C) B-1 → B HL → HL+1
OTIR	-----	16/21	ED B3	كرر OUTI إلى أن B=0
OUTD	-----	16	ED A8	B ← B-1, port (C) ← (HL) HL ← HL-1
OTDR	-----	16/21	ED B8	كرر OUTD إلى أن B=0
PUSH rp	-----	11	11rp0101	المسجلان rp ← قيمة المكدة
PUSH IY	-----	15	FD E5	المسجل IY ← قيمة المكدة
PUSH IX	-----	15	DD E5	المسجل IX ← قيمة المكدة
POP rp	-----	11	11rp0001	قيمة المكدة ← المسجلان rp
POP IY	-----	15	FD E1	قيمة المكدة ← المسجل IX
POP IX	-----	15	DD E1	قيمة المكدة ← المسجل IX
RLCA	--- CY -	4	07	CY ← A المسجل ←
RLA	--- CY -	4	17	← CY ← A المسجل ←

RRCA	--- CY -	4	0F	→ A المسجل → CY
RRA	--- CY -	4	1F	→ A المسجل → CY →
RLC reg	--- CY -	8	CB 0000sss	دوران مسجل , مثل RLCA
RLC (HL)	--- CY -	15	CB 06	دوران عنوان في HL مثل RLCA
RLC (IY+d)	--- CY -	23	FD CB dd 06	دوران عنوان (IY+d) مثل RLCA
RLC (IX+d)	--- CY -	23	DD CB dd 06	دوران عنوان (IX+d) مثل RLCA
RL (HL)	--- CY -	15	CB 16	دوران (HL) لليسار من خلال علم الحمل
RL (IX+d)	--- CY -	23	DD CB dd 16	دوران (IX+d) لليسار من خلال علم الحمل
RL (IY+d)	--- CY -	23	FD CB dd 16	دوران (IY+d) لليسار من خلال علم الحمل
RL reg	--- CY -	8	CB 00010sss	دوران reg لليسار من خلال علم الحمل
RLD	-----	18	ED 6F	دوران 4 بيت الأولى من A للشمال مع العنوان الموجود في HL
RRD	-----	18	ED 67	دوران 4 بيت الأولى من A لليمين مع العنوان الموجود في HL
RR (HL)	--- CY -	15	CB 1E	دوران (HL) لليمين من خلال علم الحمل
RR (IX+d)	--- CY -	23	DD CB dd 1E	دوران (IX+d) لليمين من خلال علم الحمل
RR (IY+d)	--- CY -	23	FD CB dd 1E	دوران (IY+d) لليمين من خلال علم الحمل
RR reg	--- CY -	8	CB 00011sss	دوران reg لليمين من خلال علم الحمل
RRC (HL)	--- CY -	15	CB 0E	دوران (HL) لليمين مثل RRCA
RRC (IX+d)	--- CY -	23	DD CB dd 0E	دوران (IX+d) لليمين مثل RRCA
RRC (IY+d)	--- CY -	23	FD CB dd 0E	دوران (IY+d) لليمين مثل RRCA
RRC reg	--- CY -	8	CB 00001sss	دوران reg لليمين مثل RRCA
RET	-----	10	C9	عودة من برنامج فرعي
RET Z	-----	5/11	C8	عودة مشروطة بعلم الصفر=1
RET NZ	-----	5/11	C0	عودة مشروطة بعلم الصفر=0
RET C	-----	5/11	D8	عودة مشروطة بعلم الحمل=1
RET NC	-----	5/11	D0	عودة مشروطة بعلم الحمل=0
RET PO	-----	5/11	E0	عودة مشروطة بعلم باريتي=0
RET PE	-----	5/11	E8	عودة مشروطة بعلم باريتي=1
RET M	-----	5/11	F8	عودة مشروطة بعلم إشارة=1
RET P	-----	5/11	F0	عودة مشروطة بعلم إشارة=0
RETI	-----	14	ED 4D	عودة من مقاطعة
RETN	-----	14	ED 45	عودة من مقاطعة ذات قناع
RES b,reg	-----	8	CB 10bbbsss	صفر في البت b في reg
RES b,(HL)	-----	15	CB 10bbb110	صفر في البت b في (HL)

RES b,(IX+d)	-----	23	DD CB dd 10bbb110	صفر في البت b في (IX+d)
RES b,(IY+d)	-----	23	FD CB dd 10bbb110	صفر في البت b في (IY+d)
RST n	-----	11	11nnn111	إعادة تشغيل
SUB reg	Z S P CY HC	4	10010sss	A ← A - reg
SUB (HL)	Z S P CY HC	7	96	A ← A - (HL)
SUB data8	Z S P CY HC	7	D6 data8	A ← A - data8
SUB (IX+d)	Z S P CY HC	19	DD 96 dd	A ← A - (IX+d)
SUB (IY+d)	Z S P CY HC	19	FD 96 dd	A ← A - (IY+d)
SBC reg	Z S P CY HC	4	10011sss	A ← A - CY - reg
SBC (HL)	Z S P CY HC	7	9E	A ← A - CY - (HL)
SBC data8	Z S P CY HC	7	DE data8	A ← A - CY - data8
SBC (IX+d)	Z S P CY HC	19	DD 9E dd	A ← A - CY - (IX+d)
SBC (IY+d)	Z S P CY HC	19	FD 9E dd	A ← A - CY - (IY+d)
SBC HL, rp	Z S P CY HC	15	ED 01rp0010	HL ← HL - CY - rp
SLA (HL)	Z S P CY HC	2	CB 26	إزاحة (HL) لليسار ، حمل 0 في بت 0
SLA (IX+d)	Z S P CY HC	4	DD CB dd 26	إزاحة (IX+d) لليسار ، حمل 0 في بت 0
SLA (IY+d)	Z S P CY HC	4	FD CB dd 26	إزاحة (IY+d) لليسار ، حمل 0 في بت 0
SLA reg	Z S P CY HC	2	CB 00100sss	إزاحة reg لليسار حمل 0 في بت 0
SRL (HL)	Z S P CY HC	2	CB 3E	إزاحة (HL) لليمين ، حمل 0 في بت 7
SRL (IX+d)	Z S P CY HC	4	DD CB dd 3E	إزاحة (HL) لليمين ، حمل 0 في بت 7
SRL (IY+d)	Z S P CY HC	4	FD CB dd 3E	إزاحة (HL) لليمين ، حمل 0 في بت 7
SRL reg	Z S P CY HC	2	CB 00111sss	إزاحة reg لليمين ، حمل 0 في بت 7
SRA (HL)	Z S P CY HC	2	CB 2E	إزاحة (HL) لليمين ، بت 7 تظل كما هي
SRA (IX+d)	Z S P CY HC	4	DD CB dd 2E	إزاحة (IX+d) لليمين ، بت 7 تظل كما هي
SRA (IY+d)	Z S P CY HC	4	FD CB dd 2E	إزاحة (IY+d) لليمين ، بت 7 تظل كما هي
SRA reg	Z S P CY HC	2	CB 00101sss	إزاحة reg لليمين،بت7تظل كما هي
SET b,reg	Z S P CY HC	8	CB 11bbbsss	واحد في البت b في reg
SET b,(IY+d)	Z S P CY HC	23	FD CB dd 11bbb110	واحد في البت b في (IY+d)
SET b,(HL)	Z S P CY HC	15	DD CB dd 11bbb110	واحد في البت b في (HL)
SET b,(IX+d)	Z S P CY HC	23	DD CB dd 11bbb110	واحد في البت b في (IX+d)
XOR reg	Z S P 0 0	4	10101sss	A ← A XOR reg

XOR (HL)	Z S P 0 0	7	AE	A ← A XOR (HL)
XOR data8	Z S P 0 0	7	EE data8	A ← A XOR data8
XOR (IY+d)	Z S P 0 0	19	FD AE dd	A ← A XOR (IY+d)
XOR (IX+d)	Z S P 0 0	19	DD AE dd	A ← A XOR (IX+d)

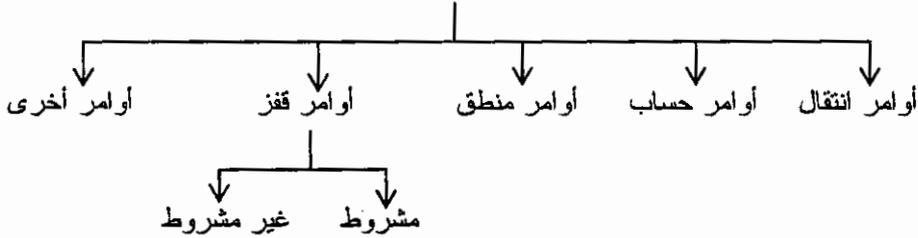
شكل (5-21) أوامر الشريحة Z80 مرتبة أبجدياً

- sss أو ddd تستبدل بشفرة مسجل 8 بت كما في فصل 2 .
- rp تستبدل بشفرة زوج مسجلات كما في فصل 2 .
- data8 ثابت مكون من 8 بت (بايت) .
- dd إزاحة عنوانية مكونة من 8 بت (بايت) .
- bbb رقم بت معينة في مسجل أو بايت ذاكرة .
- no.8 رقم (عنوان) بوابة (إدخال أو إخراج) من 8 بت (بايت) .

10-5 تمارين

1. أكمل الجدول التالي الخاص بأوامر الشريحة Z80 :

أوامر الشريحة Z80



2. ما هي نتيجة تنفيذ البرنامج التالي :

```

E000 LD A,05
E002 LD B,A
E003 LD C,B
E004 LD D,C
E005 LD E,D
E006 LD L,E
E007 LD H,L
E008 LD (HL),L
  
```

3. اقرأ البرنامج السابق وأجب عما يلي :

- محتويات مكان الذاكرة E000=.....
- محتويات مكان الذاكرة E001=.....

- محتويات مكان الذاكرة=0505
- 4.

E000 LD HL,E100
 E003 LD (HL),3E
 E005 INC HL
 E006 LD (HL),05
 E008 INC HL
 E009 LD (HL),47
 E00B INC HL
 E00C LD (HL),48

- ما هي نتيجة تنفيذ البرنامج السابق ؟
5. على ضوء نتيجة تنفيذ البرنامج السابق ما هي نتيجة تنفيذ الشفرات الموجودة في الأماكن E100 إلى E103 ؟
 6. هل تتأثر الأعلام بأوامر الانتقال ؟
 7. أذكر الأعلام التي تتأثر بكل عملية من العمليات الحسابية والمنطقية؟
 8. إذا كانت محتويات المسجل A=F3H ومحتويات المسجل B=A4H فاكتب محتويات المسجل A بعد تنفيذ كل أمر من الأوامر التالية على نفس المحتويات السابقة ووضح أيضا كيف ستتأثر الأعلام بكل أمر :

ADD A,B
 SUB A,B
 SUB A,A
 INC A
 AND B
 OR B
 XOR B

9. ارسم مخطط السير للبرنامج التالي وما هي نتيجة تنفيذه :

E000 LD L,50H
 E003 LD H,E1H
 E005 LD (HL),A
 E006 DEC L
 E007 JPNZ E005

10. ماذا يحدث لو كتبنا البرنامج السابق عند E100 بدلا من E000 ؟
11. أعد كتابة البرنامج السابق مستخدما العلامات Labels ؟ وما هي مميزات البرنامج مكتوبا بهذه الصورة ؟
12. اكتب برنامجا يحسب عدد الواحد الموجودة في محتويات المسجل A , مثلا إذا كان A=11110101 فإن عدد الواحد = 6 .

13. كم عدد بوابات الإدخال التي يستطيع المعالج Z80 التعامل معها؟
14. كم عدد بوابات الإخراج التي يستطيع المعالج Z80 التعامل معها؟
15. على ماذا يتوقف هذا العدد ؟
16. هل هناك ما يمنع أن تكون بوابتي إدخال وإخراج لهما نفس الرقم ،
كمثال على ذلك IN 05 و OUT 05؟
17. OUT (C),reg هذا أحد أوامر الإخراج للمعالج Z80 والذي يعنى أخراج
محتويات المسجل reg على بوابة الإخراج التي رقمها فى المسجل C
فهل المعالج 8085 لديه ما يكافىء هذا الأمر ؟
18. هل تتأثر الأعلام بأوامر الإدخال والإخراج ؟
19. أكتب برنامجاً يقرأ محتويات البوابة 00 وإذا كانت هذه المحتويات
زوجية يخزنها فى الذاكرة ابتداء من العنوان E100 وإذا كانت فردية
يخرجها على البوابة 00 ؟
20. اذكر طرق العنونة memory addressings المستخدمة مع المعالج Z80؟
والأوامر المستخدمة مع كل طريقة ؟ ومتى يفضل استخدام كل طريقة ؟
21. اكتب برنامج يحسب أكبر قيمة عددية فى بايت فى المدى العنوانى
E200H إلى E250H .
22. اكتب برنامج يحسب عدد البايتات التى تحتوى أصفراً والتي تحتوى
أرقاماً موجبة والنى تحتوى أرقاماً سالبة فى المدى العنوانى E100 إلى
E150 .
23. اكتب برنامج يحسب عدد البايتات التى تحتوى بيانات فردية والتي
تحتوى بيانات زوجية فى المدى العنوانى E100 إلى E150 .
24. المدى العنوانى E100 إلى E150 يحتوى بيانات لإشارة صوت ، احسب
كم مرة عبرت إشارة الصوت الصفر .
25. اكتب برنامج يقرأ بوابة الإدخال رقم 00 ويختبر البت الرابعة فيها ، فإذا
كانت هذه البت واحد يخرج هذه المحتويات على البوابة 00 ، وإذا كانت
هذه البت صفر يخرج محتوياتها على البوابة 01 .
26. اكتب برنامج يقرأ بوابة الإدخال رقم 00 إلى مالانهاية ويختبر البيانات
التي يقرأها ، فإن كانت فردية يخرجها على البوابة 00 ، وإن كانت
زوجية يخرجها على البوابة 01 . احسب أكبر معدل لدخول البيانات لكى
يعمل هذا النظام فى الزمن المباشر real time .