

لوحة المفاتيح



بإشراف الدكتور

حسن الصالح

مادة نظم الزمن الحقيقي

حلب

السبت، 01 كانون الثاني، 2005

إعداد

أريج حاج محرم

ألآء الحطاب

رنا غبشة

مروان الريحاوي

الفهرس

4	1	مقدمة
5	1.1	أساسيات برمجة لوحة المفاتيح
5	1.2	تحويل الـ SCAN CODE
6	1.3	شيفرات العمل و لقطع
6	1.4	معالج الـ ROM-BIOS في لوحة المفاتيح
7	1.5	التحويل من شيفرة المسح إلى شيفرة الـ ASCII :
7	1.6	استخدام لغات أخرى في لوحات المفاتيح:
7	1.7	لوحات مفاتيح الحواسيب الشخصية:
7	1.8	لوحات مفاتيح الـ XT و PC :
7	1.9	لوحة مفاتيح الـ MF-II :
8	1.10	شيفرات التحكم
9	1.11	شيفرة الـ ASCII :
10	1.12	شيفرات لوحة المفاتيح الموسعة :
13	2	توايح المقاطعة 16H في الـ BIOS :
13	2.1	التابع 00H (قراءة لوحة المفاتيح)
13	2.2	التابع 01H (قراءة لوحة المفاتيح)
13	2.3	التابع 02H (قراءة مفاتيح التحكم)
14	2.4	برنامج الـ KEYP.PAS :
15	2.5	توايح جديدة في الـ BIOS :
20	3	برنامج خدمة مقاطعة لوحة المفاتيح :
21	3.1	الوصول إلى المقاطعة 16H
21	3.2	مثال :
21	3.3	برنامج خدمة مقاطعة جديد.
23	4	التحكم بالمقاطعة الصلبة للوحة المفاتيح :
26	5	برمجة المتحكم بلوحة المفاتيح :
30	6	أمثلة تطبيقية :
30	6.1	مثال تغيير معدل التكرار و سرعة لوحة المفاتيح :
31	6.2	مثال التحكم بأضواء الإشارة على لوحة المفاتيح من النوع AT.
32	6.3	تطبيق تغيير أشكال الأحرف المطبوعة (من أجل اللغة العربية)
34	7	الملاحق
34	7.1	جدول شيفرات المسح :
36	7.2	الملحق 1.
36	7.3	الملحق 2
37	8	المراجع المعتمدة :

بسم الله الرحمن الرحيم

1 مقدمة

تعتبر لوحة المفاتيح الجهاز المحيطي الأساسي المستخدم في التعامل مع الحاسب الشخصي و إدخال البيانات النصية و الرقمية إليه، و انطلاقاً من أهميته سنتناول في دراستنا هذه الجهاز من ناحية ربطه مع الحاسب الشخصي.

و سنبحث في هذه الدراسة في النقاط التالية :

- أساسيات و تصميمات لوحة المفاتيح.
- المقاطعات الموظفة لتخديم هذه المحيطية
- كيفية التعامل مع المقاطعات السابقة و برمجتها
- التحكم بلوحة المفاتيح من خلال إرسال أوامر لمعالجها الصغرى.

كما قدمنا في هذه الدراسة العديد من الأمثلة الشارحة للفرات النظرية و كما حاولنا التعاون مع مجموعة الـ VGA للقيام بمحاولة تعريب المعارف الموجودة بالـ BIOS من خلال مثال لهذا يوضع الآلية للبداية في هذا الموضوع.

راجين الله تعالى أن نكون قد أنجزنا ما علينا من واجب و آمليين بمتابعة هذا المشروع و متابعة أفكاره التطويرية المقترحة، و للمراجعة حول أي نقطة عن الموضوع يمكنكم الاتصال مع أحد أفراد المجموعة التي نفذت هذه الدراسة من خلال :

- مروان الريحاوي : mar-rih@myway.com
- مروان الريحاوي : mar-rih@mail.sy
- أو من خلال الموقع التالي : www.alepposoft.com/info
- أو من خلال هاتف مروان الريحاوي التالي : 021-5222188

مجموعة لوحة المفاتيح
حلب
2005/01/08

إن لوحة المفاتيح هي جهاز الدخل الأساسي للكمبيوتر و هي أكثر أجهزة الدخل استخداما من قبل DOS .

إن برامج الـ TSR (البرامج المقيمة في الذاكرة) تتعامل كثيرا مع لوحة المفاتيح مثل برامج خدمة المقاطعة المتعلقة بتغيرات لوحة المفاتيح.

في هذا البحث سنشرح آلية حدوث هذه المقاطعات وكيف نبرمج لوحة المفاتيح بشكل مباشر وسوف نختبر تأثير هذه البرمجة على طريقة عمل لوحة المفاتيح مثلا تستطيع التحكم بإضاءة الثنائيات الضوئية في لوحة المفاتيح .

1.1 أساسيات برمجة لوحة المفاتيح

عندما يضغط المستخدم مفتاح من لوحة المفاتيح تتشكل نبضة كهربائية تعرف موقع المفتاح و تتم معالجة هذه الإشارة من قبل معالج لوحة المفاتيح الذي يتوضع داخل اللوحة .

عادة يكون هذا المعالج شريحة 8048 INTEL أو ما يقابله من انتاج آخر .

في الكمبيوترات من الصنف AT يعالج الاتصال بواسطة شريحة INTEL 8042

و بواسطتها فإن هذه الكمبيوترات قادرة على الاتصال ثنائي الاتجاه بين الـ CPU و لوحة المفاتيح علما أن الكمبيوترات الأقدم لا تمتلك هذه الخاصة .

1.2 تحويل الـ SCAN CODE

يحول معالج لوحة المفاتيح النبضات الكهربائية المحددة بموقع المفتاح إلى عدد يدعى بالـ SCAN CODE (شيفرة المسح) و لا توجد علاقة بين هذه الشيفرة و الحرف الموجود على المفتاح المضغوط أو الوظيفة التي يؤديها المفتاح في البرنامج الحالي .

يمرر معالج لوحة المفاتيح شيفرة المسح إلى الكمبيوتر في الـ AT يستقبل المتحكم في لوحة المفاتيح الرمز و هذا النقل يكون بشكل تسلسلي ما دام الكابل الواصل بين الكمبيوتر و لوحة المفاتيح يملك خط معطيات وحيد و هذا الاتصال متزامن على عكس الاتصال غير المتزامن في المنفذ التسلسلي .

يتحقق الاتصال المتزامن باستخدام خط الساعة و خط المعطيات .

ينقل خط الساعة إشارات مؤقتة بواسطة تقطيع مستمر من المرتفع إلى المنخفض (الصفير و الواحد). و يتم مزامنة نقل بتات شيفرة المسح المنفصلة وفقا لنبضة الساعة تلك .

إذا تم ضغط عدة مفاتيح معا فإن المعالج يخزنها في ذاكرة داخلية و تحوي هذه الذاكرة مساحة فارغة تكفي لعشر ضربات مفاتيح و الحقيقة أن المعطيات تمر إلى المعالج بسرعة أكبر بكثير من سرعة طباعة المستخدم.

1.3 شيفرات العمل و القطع

تشكل شيفرة المسح أيضا عند تحرير المفتاح و هذا يحدد للنظام إذا كان المفتاح ما يزال مضغوطا أو تم تحريره للتو و هذا مهم جدا لأنها الطريقة الوحيدة التي يستطيع فيها الكمبيوتر أن يترجم و بشكل صحيح الحالة عندما يتم ضغط أكثر من مفتاح في وقت واحد و بدون هذه الخاصة لن يكون بالإمكان تنفيذ مهام خاصة مثل طباعة الأحرف الكبيرة أو إعادة تشغيل الكمبيوتر بواسطة CTRL+ALT+DEL .

يستخدم النظام ما يسمى شيفرات العمل و القطع للتمييز بين شيفرات المسح للمفاتيح المضغوطة (العمل) و المفاتيح التي تحررت (القطع) الفرق الوحيد بينها هو البت 7 الذي يكون بقيمة 1 في شيفرة القطع و هذا يؤدي إلى أمرين مهمين :

- الأول: شيفرات القطع دائما أكبر من 128 و شيفرات العمل أصغر منها .
- الثاني : لا يمكن للوحة المفاتيح أن تحوي أكثر من 128 مفتاح وإلا سوف تتجاوز شيفرات العمل شيفرات القطع .

المثال الأكثر وضوحا عن ضغط أكثر من مفتاح معا هو كتابة الحرف الكبير فمثلا لكتابة الحرف A على المستخدم أن يستمر بالضغط على SHIFT اليميني و بعد ذلك الضغط على A معالج اللوحة يمرر شيفرة العمل لـ SHIFT وهي (36H) ثم شيفرة العمل للحرف A و هي (1EH) للكمبيوتر و طالما لم يتسام النظام شيفرة القطع لـ SHIFT يفترض أن المفاتيح مضغوطة معا و يولد حرف كبير بدلا من الصغير .

1.4 معالج الـROM-BIOS في لوحة المفاتيح

كيف يستقبل المعالج شيفرات المسح ؟

تتنفذ المقاطعة الصلبة IRQ1 في كل مرة ترسل فيها لوحة المفاتيح شيفرة عمل أو قطع إلى الكمبيوتر و هذا بدوره يستدعي المقاطعة 09H و يستقبل الروتين المسؤول عن معالج لوحة المفاتيح شيفرات العمل و القطع و يحولها إلى شيفرات ASCII الموافقة التي يمكن قرائتها من قبل التطبيق الحالي .

هنالك عدة مهام يجب إنجازها قبل أن يتمكن التطبيق من قراءة المفاتيح :

الأولى : يجب على معالج لوحة المفاتيح أن يقرأ شيفرة العمل أ و القطع من اللوحة باستخدام منفذ دخل/خرج و عنوان هذا المنفذ هو 60H في كل أنظمة الحواسيب و يقرأ هذا المنفذ شيفرات العمل و القطع فقط . يقدر معالج لوحة المفاتيح الشيفرات و يقرر إذا كان قد أدخل المحرف . وكما رأينا ليست كل نتائج الضغط على المفاتيح هي مرئية على الشاشة فمثلا الحرف A لم يظهر إلا بعد الضغط على الحرف نفسه و حالما يتعرف المعالج على المحرف المدخل يقوم بتحويل المحرف إلى الشيفرة التي يستطيع التطبيق الحالي فهمها , شيفرات المسح بحد

ذاتها غير مستخدمة لأن لوحات المفاتيح المختلفة تستخدم مجموعات مختلفة من شيفرات المسح مع أن معظم المجموعات متشابهة .

1.5 التحويل من شيفرة المسح إلى شيفرة الـ ASCII :

تحول شيفرة المسح إلى شيفرة الـ ASCII و التي هي قياسية في كل الحواسيب , حيث تتألف شيفرة الـ ASCII من 128 حرف وهناك مجموعة أخرى مؤلفة من 256 حرف .

لا يمرر حرف الـ ASCII المحول مباشرة إلى التطبيق بل يخزن أولاً في buffer خاص لهذا الغرض ثم يقوم بعدئذ معالج لوحة المفاتيح بإتمام عمله ثم يقوم التطبيق بقراءة هذه المحارف من هذا الـ buffer ليقوم بمعالجتها . المقاطعة H 16 من الـ ROM-BIOS لها عدة توابع متاحة لهذا الغرض .

1.6 استخدام لغات أخرى في لوحات المفاتيح:

على الرغم من أن الـ ROOMBIOS يعرف معالج لوحة المفاتيح المستخدم إلا أن الـ DOS يستطيع أن يبدله ببرنامج آخر و بشكل قياسي يعتمد اللغة الأنكليزية و لاستخدام محارف من لغات أخرى يمكن تنصيبها في ملف الـ AUTOEXEC.BAT لاستخدامها .

1.7 لوحات مفاتيح الحواسيب الشخصية:

يوجد ثلاثة أنواع قياسية للوحات مفاتيح الحواسيب الشخصية وهي :

لوحة المفاتيح المنتجة من قبل الـ IBM وتمتاز هذه اللوحة بأنها لوحة مفاتيح قياسية و أشكالها متنوعة لأنها تصمم بلغات مختلفة عديدة حيث يمكن لمفاتيحها أن تتوضع في أمكنة مختلفة و تحتوي على رموز مختلفة أيضاً ولكن هذه الرموز لا تدل على ما يحدث داخل اللوحة .

1.8 لوحات مفاتيح الـ XT و PC :

كان الحاسوب في البداية يرفق بلوحة مفاتيح من النوع PC/XT و المؤلفة من 83 مفتاح . حيث كان مفتاح الـ Enter و مفاتيح الـ Shift صغيرة و صعبة الاستخدام .

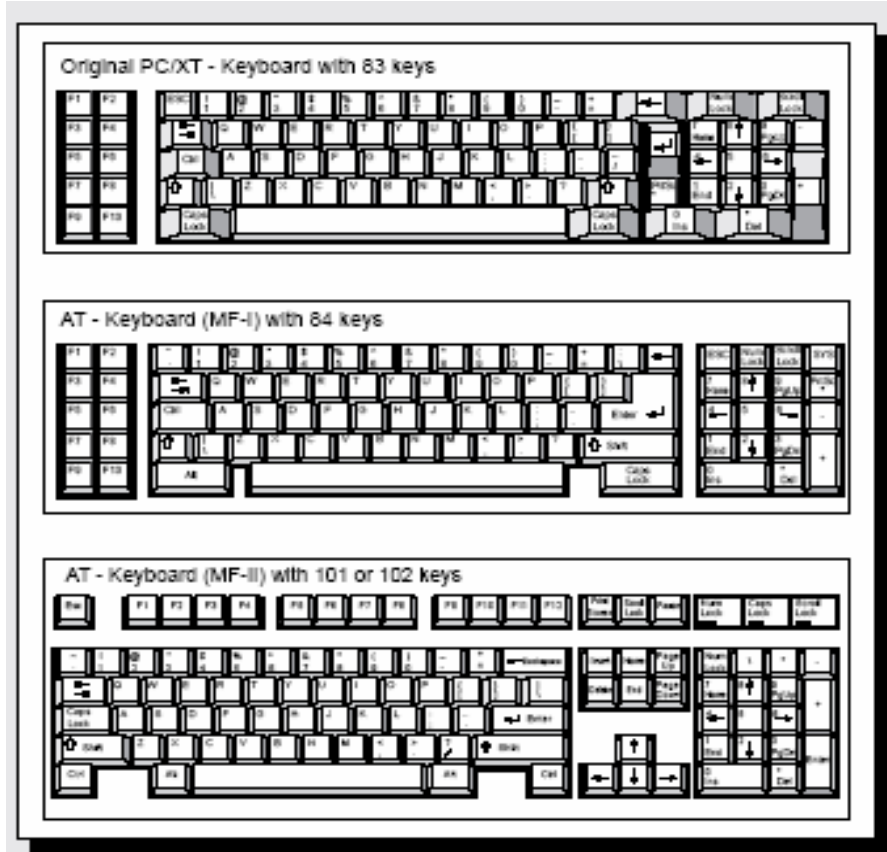
ثم تم انتاج لوحة مفاتيح الـ AT التي حلت هذه المشكلة بجعل مفتاح الـ Enter و مفاتيح الـ Shift أكبر و من السهل ايجادها . ولكن مع ذلك كانت تحتوي على مفاتيح صغيرة الحجم مثل الـ Num Lock و الـ Scroll Lock . لأن هذه المفاتيح لا تستخدم بشكل متكرر مثل مفاتيح الـ Shift .

1.9 لوحة مفاتيح الـ MF-II :

وهي تحتوي على ما يلي :

■ مجموعة من مفاتيح أسهم مخصصة منفصلة عن الـ keypad .

- مفاتيح وظيفية متوضعة في أعلى لوحة المفاتيح.
- مفاتيح الـ Alt و التي تتوضع في أسفل اللوحة ويمكن الوصول إليها بسهولة.
- الـ LEDs الثلاثة التي تحدد الحالة وهي Num Lock و Scroll Lock و Caps Lock .



الشكل 1-1

1.10 شيفرات التحكم

كما نعلم سابقاً أنه أي محرف من محارف الـ Ascii يمكن أن يتم ادخاله من خلال لوحة المفاتيح باستخدام مفتاح الـ ALT و مفاتيح لوحة المفاتيح الرقمية كما أنه من الممكن استخدام مفتاح الـ CTRL لتلك الوظيفة أيضاً. و عند استخدام هذه المفاتيح فإنه يمكن إظهار احرف الـ ASCII التي شيفرتها هي أصغر من الـ 32 و الشكل التالي يبين ما هي هذه المفاتيح التي سوف تظهر.

Dec	Symbol	Keyboard codes	Dec	Symbol	Keyboard codes
0	Empty (Null)	Ctrl + Z	16	▶	Ctrl + P
1	☺	Ctrl + A	17	◀	Ctrl + O
2	☹	Ctrl + B	18	↕	Ctrl + R
3	♥	Ctrl + C	19	!!	Ctrl + S
4	♦	Ctrl + D	20	¶	Ctrl + T
5	♣	Ctrl + E	21	⓪	Ctrl + U
6	♠	Ctrl + F	22	▬	Ctrl + V
7	•	Ctrl + G	23	↕	Ctrl + W
8	•	Ctrl + H BS Shift + Backsp	24	↑	Ctrl + X
9	○	Ctrl + I TAB	25	↓	Ctrl + Y
10	●	Ctrl + J LF Ctrl + Enter	26	→	Ctrl + Z
11	♂	Ctrl + K	27	←	Ctrl + Esc ESC Esc Shift + Esc
12	♀	Ctrl + L	28	└	Ctrl + [
13	♪	Ctrl + M CR Shift + Enter	29	↔	Ctrl +]
14	♫	Ctrl + N	30	▲	Ctrl + ^
15	⊕	Ctrl + O	31	▼	Ctrl + _
			32	Space	Spacebar Shift + Spacebar Ctrl + Spacebar Alt + Spacebar

الشكل 2-1

1.11 شيفرة الـ ASCII :

إذا احتوى المسجل AL على قيمة مختلفة عن الصفر فهي قيمة شيفرة الـ ASCII و نجد الـ SCAN CODE في المسجل AH و ذلك للمفتاح المفعّل حالياً مع بعض الاختلافات بالنسبة لمفاتيح التحكم و على سبيل المثال فإن معظم البرامج تترجم الرمز 27 على أنه تعليمة هروب بدلاً من الرغبة في إدخال المحرف كنص للكتابة .

و لحل المشكلة يمكن استخدام مفتاح وظيفي مثل F1 للإشارة الى رغبتنا في استخدام هذا المفاتيح في النصوص حيث يتم فحص المفتاح F1 إذا كان مضغوطاً يعتبر ما بعده نص للكتابة و إلا يستخدم كمفتاح للهروب.

1.12 شيفرات لوحة المفاتيح الموسعة :

ASCII control codes on PCs			
Code	Meaning	Code	Char.
8	Backspace	BS	
9	Tab	TAB	
10	Linefeed (Ctrl + Enter)	LF	XXX
13	Carriage Return	CR	
27	Escape	ESC	

جدول 1

تقدم توابع BIOS شيفرات لوحة المفاتيح الموسعة أيضا و لكن في هذه الحالة نجد 00 في المسجل AL و قيمة الـ SCANE CODE في المسجل AH و لذلك فإنه عند استدعاء أحد التوابع 00H, 01H, الخاصة بالمقاطعة 16H للـ BIOS سيجد البرنامج المستدعي القيمة 00 في AL و بهذه الطريقة نستخدم 256 شيفرة جديدة .

Extended key codes			
Code (hex)	Code (dec)	Key(s)	
0FH	15	Shift	Tab
10H	16	Alt	Q
11H	17	Alt	W
12H	18	Alt	E
13H	19	Alt	R
14H	20	Alt	T
15H	21	Alt	Y
16H	22	Alt	U
17H	23	Alt	I
18H	24	Alt	O
19H	25	Alt	P
1EH	30	Alt	A

جدول 2

Extended key codes			
Code (hex)	Code (dec)	Key(s)	
1FH	31	Alt	S
20H	32	Alt	D
21H	33	Alt	F
22H	34	Alt	G
23H	35	Alt	H
24H	36	Alt	J
25H	37	Alt	K
26H	38	Alt	L
2CH	44	Alt	Z
2DH	45	Alt	X
2EH	46	Alt	C
2FH	47	Alt	V
30H	48	Alt	B
31H	49	Alt	N
32H	50	Alt	M
03BH	59	F1	
3CH	60	F2	
3DH	61	F3	
3EH	62	F4	
3FH	63	F5	
40H	64	F6	
41H	65	F7	
42H	66	F8	
43H	67	F9	
44H	68	F10	
47H	71	Home	
48H	72	↑	
49H	73	Page Up	
4BH	75	←	
4DH	77	↔	
50H	80	↓	
51H	81	Page Dn	
52H	82	Insert	
53H	83	Del	
54H	84	Shift	F1
55H	85	Shift	F2

جدول 3

Extended key codes			
Code (hex)	Code (dec)	Key(s)	
56H	86	Shift	F3
57H	87	Shift	F4
58H	88	Shift	F5
59H	89	Shift	F6
5AH	90	Shift	F7
5BH	91	Shift	F8
5CH	92	Shift	F9
5DH	93	Shift	F10
5EH	94	Ctrl	F1
5FH	95	Ctrl	F2
60H	96	Ctrl	F3
61H	97	Ctrl	F4
62H	98	Ctrl	F5
63H	99	Ctrl	F6
64H	100	Ctrl	F7
65H	101	Ctrl	F8
66H	102	Ctrl	F9
67H	103	Ctrl	F10
68H	104	Alt	F1
69H	105	Alt	F2
6AH	106	Alt	F3
6BH	107	Alt	F4
6CH	108	Alt	F5
6DH	109	Alt	F6
6EH	110	Alt	F7
6FH	111	Alt	F8
70H	112	Alt	F9
71H	113	Alt	F10
73H	115	Ctrl	←
74H	116	Ctrl	→
75H	117	Ctrl	End
76H	118	Alt	
77H	119	Alt	Home
78H	120	Alt	1
79H	121	Alt	2
7AH	122	Alt	3

جدول 4

Extended key codes			
Code (hex)	Code (dec)	Key(s)	
7BH	123	Alt	4
7CH	124	Alt	5
7DH	125	Alt	6
7EH	126	Alt	7
7FH	127	Alt	8
80H	128	Alt	9
81H	129	Alt	0
82H	130	Alt	
83H	131	Alt	'

جدول 5

2 توابع المقاطعة 16H في الـ BIOS :

2.1 التابع 00H (قراءة لوحة المفاتيح)

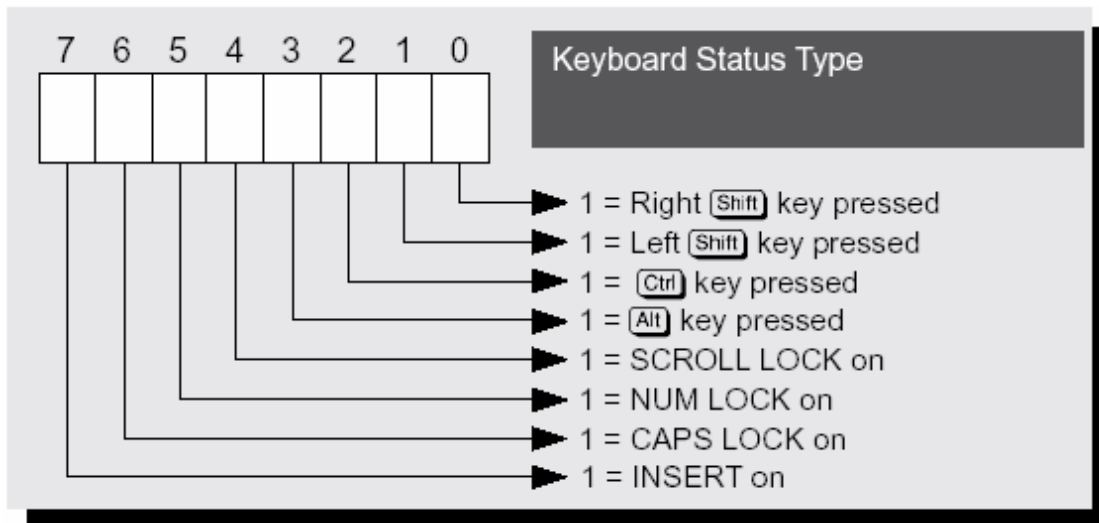
عندما تحدث المقاطعة 16H فإذا كان المحرف موجودا في الـ BUFFER فإن يتم حذفه منها و تمريره إلى البرنامج المستدعي أما إذا كانت الـ BUFFER فارغة يتم انتظار الإدخال ثم إعادة القيمة إلى البرنامج .

2.2 التابع 01H (قراءة لوحة المفاتيح)

يقوم هذا التابع أيضا بقراءة لوحة المفاتيح مثل التابع السابق ولكن على عكس التابع 00H لا يتم حذف المحرف من الـ BUFFER ويتم إعلام البرنامج عن طريق علم التصفير (ZEROFLAG) فإذا كان مساويا للواحد لا يوجد محرف و في الحالة المعاكسة يحوي كل من AH,AL على معلومات عن المفتاح الذي تم ضغطه

2.3 التابع 02H (قراءة مفاتيح التحكم)

لهذا التابع مهمة مختلفة فهو يقوم بقراءة حالة مفاتيح التحكم حيث نضع 02H في المسجل AH و نجد شعاع الحالة في المسجل AL بعد الاستدعاء و على سبيل المثال إذا كان البت الثالث في الشعاع مساويا للواحد فإن مفتاح ALT مضغوط علما ان شعاع الحالة هذا مفيد في برامج TSR التي تفحص حالة المفاتيح باستمرار.



الشكل 1-2

(دمج المفاتيح الموسعة عند استدعاء التوابع 01H-11H)

2.4 برنامج KEYP.PAS :

يقوم هذا البرنامج بعدد من المهام فهو يعرض على زاوية الشاشة اليمنى حالة المفاتيح :

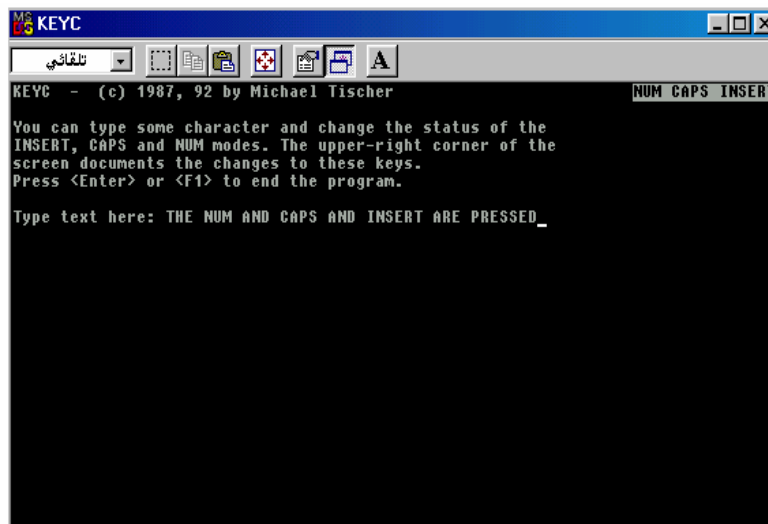
INSERT , CAPSLOCK , NUMLOCK

و هذا مهم في حالة لوحة المفاتيح XT لأنها لا تحوي على ديودات كما في لوحة المفاتيح AT و بالتالي لا يمكننا معرفة الوضع الحالي لهذه المفاتيح

يقوم البرنامج بقراءة شعاع الحالة بواسطة التابع 02H و يتم تهيئة ثلاث أعلام تراقب حالة هذه البتات و تغيرها على الشاشة لتتوافق مع حالتها

بالإضافة إلى هذه المهمة يقوم البرنامج بقراءة لوحة المفاتيح باستخدام التابع 01H حيث يتم فحص وجود محرف في الـ BUFFER و في حالة عدم وجوده يقفز إلى أول البرنامج ليفحص شعاع الحالة من جديد و ذلك ضمن حلقة لإظهار كل التغيرات على الشاشة أما الخطوة الأخيرة في البرنامج هي فحص شيفرة المفاتيح الموسعة حيث يضيف البرنامج 256 إلى الشيفرة لإعلام التابع المستدعي بوجود مفتاح موسع .

في الشكل التالي تنفيذ للبرنامج يظهر الوضع الراهن للمفاتيح في الزاوية اليمينية العليا و إمكانية كتابة نص و عرضه على الشاشة



الشكل 2-2

....\PROGRAMS\C

/PROGRAMPAS

ملاحظة : البرنامج موجود في المجلد

الملف : TYPMC.C

كما انه موجود بلغة الباسكال في المجلد

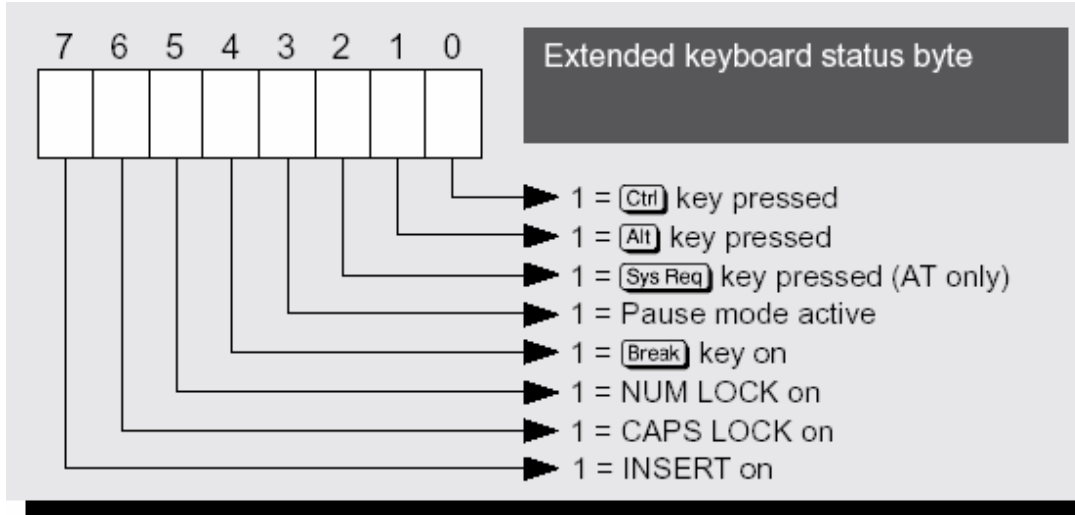
الملف KEYP.PAS

2.5 توابع جديدة في الـ BIOS :

تعرف هذه التوابع بالأرقام 10H , 11H, 12H و تعمل بنفس الأسلوب الذي تعمل به التوابع 00H , 01H , 02H .

الفرق بين التوابع 01H , 00H و نظيراتها يتضمن الشيفرة التي يتم إرجاعها و هذا يطبق بشكل خاص لتوسيع شيفرات لوحات المفاتيح الغير موجودة في XT

تمثل التوابع 11H , 10H نظائرها و يختلف التابع 12H عن 02H في النتيجة المرجعة فالتابع الجديد يعيد معلومات مختلفة تعرف بشعاع الحالة الموسع و يحتوي على معلومات عن الوضع الحالي للمفاتيح الوظيفية بالإضافة إلى مفاتيح CTRL , ALT اليساري و اليميني و التي لا يقرؤها شعاع الحالة القديم .



الشكل 3-2

مثال:

يقوم هذا البرنامج بقراءة لوحة المفاتيح باستخدام تابع الـ BIOS (10H) وعرض شيفرة الـ ASCII والـ SCAN المقابلة للحرف المدخل من لوحة المفاتيح , فهو يعرض الشيفرة المقابلة للمفاتيح الخارجية ومجموعة المفاتيح المتوفرة على لوحة المفاتيح MF_II .

حيث يمكن للمستخدم ببساطة مقارنة توابع الـ BIOS الموسعة مع التوابع العادية و سيستمر هذا التنفيذ حتى يقوم المستخدم بضغط المفتاح . ESC

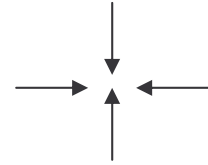
```

C:\WINDOWS\System32\cmd.exe - mf2c
MF2C - (c) 1992 by Michael Tischer
BIOS functions implemented for MF-II keyboards.
Press any key or combination to display key codes.
Press <Esc> to end the program.
Scan : 02 ASCII: 31
Scan : 03 ASCII: 32
Scan : 04 ASCII: 33
Scan : 05 ASCII: 34
Scan : 48 ASCII: E0 <---- MF-II key
Scan : 4B ASCII: E0 <---- MF-II key
Scan : 50 ASCII: E0 <---- MF-II key
Scan : 4D ASCII: E0 <---- MF-II key

```

الشكل 4-2

الشكل السابق يظهر شيفرة الـ ASCII والـ SCAN للمحارف العادية " 1 2 3 4 "



والمحارف الموسعة على التوالي.

....\PROGRAMS\C

ملاحظة : البرنامج موجود في المجلد
الملف : MF2C.C

متحولات مقاطعات لوحة مفاتيح الـ BIOS :

تملك لوحة المفاتيح ثماني متغيرات من أجل إدارة لوحة المفاتيح والاتصال بين معالج مقاطعة لوحة المفاتيح (المقاطعة 09H) وتوابع لوحة مفاتيح الـ BIOS (المقاطعة 16H).

هذه المتحولات مفيدة من أجل برامج الـ TSR التي تغير طريقة عمل مقاطعات لوحة المفاتيح , حتى البرامج العادية يمكن أن تجد طرقا لمعالجة هذه المتحولات .

إن المستخدم متآلف مسبقا مع نوعين من هذه المتغيرات :بايت حالة لوحة المفاتيح وبايت حالة لوحة المفاتيح الموسعة .

عنوان الإزاحة (19H) هو البايت المستخدم عند إدخال شيفرة الـ ASCII مع المفتاح ALT واللوحة الرقمية. فعندما يتم ضغط مفاتيح الأرقام سيخزن الكود المدخل في هذا البايت .

ان متحولات مقاطعة لوحة مفاتيح الـ BIOS مسجلة ضمن القائمة التالية:

BIOS variables for keyboard management		
Offset	Meaning	Type
17H	Keyboard status	1 byte
18H	Extended keyboard status	1 byte
19H	Code for ASCII input	1 byte
1AH	Next character in keyboard buffer	1 word
1CH	Last character in keyboard buffer	1 word
1EH	Keyboard buffer	16 words
80H	Start address of keyboard buffer	1 word
82H	End address of keyboard buffer	1 word

جدول 6

BUFFER لوحة المفاتيح :

المتحولات الثلاثة التي تلي عناوين الإزاحة 1AH,1CH,1EH تدير مسجل لوحة المفاتيح . حيث أن معالج مقاطعة لوحة المفاتيح (المقاطعة 09H) يخزن ضغطة كل مفتاح بحيث يمكن للتطبيقات قراءتها باستخدام مقاطعة لوحة مفاتيح الـ BIOS (16H) .

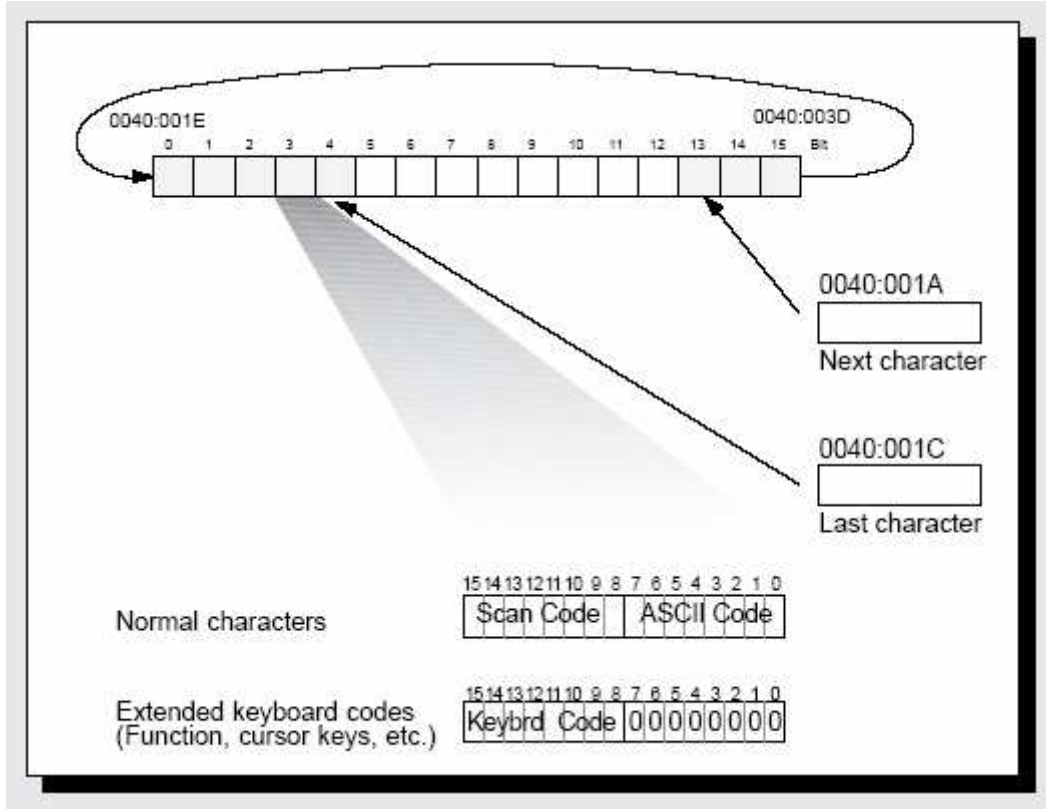
ان buffer لوحة المفاتيح مبني كـ buffer حلقي , الذي يستخدم لكتابة المحارف إلى الـ buffer وقراءتها منه بنفس الوقت .

في البداية كانت أفضل طريقة لتخزين المحارف في هذا الـ buffer هي الطريقة التسلسلية (المحرف الأول في الموقع الأول والمحرف الثاني في الموقع الثاني وهكذا...) فبعد أن تتم قراءة المحرف الأول سيمحي من موقعه الأول وتنتم إزاحة بقية المحارف الأخرى.

على الرغم من أن هذه الطريقة تعمل بشكل جيد إلا أنها تتجز عملها لا حاجة له بالنسبة للمعالج لأنه في كل مرة يقرأ فيها محرف ما , يجب أن تتحرك كل المحارف الأخرى في الـ buffer , لذلك سيستخدم الـ buffer الحلقي مؤشران: الأول يشير إلى الموقع الذي سيقراً منه المحرف التالي والثاني يشير إلى المكان الذي سيكتب فيه المحرف التالي الجديد . وبهذه الطريقة ستتحرك المؤشرات بدلا من المحتويات الداخلية للـ buffer .

في البداية يشير كلا من هذان المؤشران إلى بداية المسجل وستحركان باتجاه نهايته مع كل قراءة وكتابة لمحرف ما .
 عندما يصل كلا المؤشران إلى نهاية الـ buffer ستتم إعادتهما إلى بداية الـ buffer.
 ومن حركة المؤشر الدائرية يأتي اسم الـ buffer الحلقي.

مثال: افترض أن عنوان إزاحة المحرف التالي الذي سيقراً من buffer لوحة المفاتيح هو (1AH) حالما تتم قراءته يتحرك المؤشر بايتين باتجاه نهاية الـ buffer .



الشكل 5-2

يحدث نفس الشيء مع المؤشر (1CH) الذي يشير إلى الموقع التالي للمحرف الأخير في الـ buffer. فإذا ضغط المستخدم مفتاح آخر سيخزن في هذا الموقع عندئذ يتحرك المؤشر بايتين باتجاه نهاية الـ buffer فإذا كانت قد خزنت ضغطة مفتاح جديدة في آخر word من الـ buffer عندها سيعود المؤشر ليشير إلى بداية الـ buffer .

إذا كل محرف يحتاج بايتين ليخزن في buffer لوحة المفاتيح ,
 البايت الأول للـ ASCII CODE والثاني للـ SCAN CODE .

ان العلاقة بين مؤشري هذا الـ buffer تبين حالته التي تتحدد بالشرطين التاليين :

- اذا كان لكلا المؤشرين نفس القيمة , هذا يعني أن مسجل لوحة المفاتيح فارغ.
- اذا كان مؤشر النهاية يحاول أن يشغل نفس مكان مؤشر البداية , هذا يعني أن المؤشر مليء.

يتضمن buffer لوحة المفاتيح 32byte وبما أن كل حرف يحتاج بايتين ليخزن فيه , اذا" يمكن للـbuffer تخزين 16 حرف هذا من أجل حرف الـASCII العادي , أما من أجل كود لوحة المفاتيح الموسعة سيكون الـ ASCII CODE مساويا" للقيمة (0) لأن كود الحرف الفعلي يكون في البايت اللاحق.

موقع buffer لوحة المفاتيح :

يتوضع buffer لوحة المفاتيح في عنوان الازاحة (1EH) . وكان هذا سبب تصميم (2 buffers) حلقيان , فالمؤشران سيجعلان حجم وموقع buffer لوحة المفاتيح يتغيران عند الحاجة .

مثال: NOKEY.C

هذا البرنامج يبين كيفية معالجة متحولات الـ BIOS , ويستخدم تابع هام جدا لبرمجة لوحة المفاتيح الأساسية ولتنظيف محتوى buffer لوحة المفاتيح .

إذا كان برنامجك يحتاج لأن يحاول المستخدم حذف الملفات وتهيئة الأقراص وأنت لا تريد أية ضغطة مفتاح غير مقصودة مخزنة في buffer لوحة المفاتيح , المفتاح الخاطئ سيخبر برنامجك أن يبدأ قبل أن يملك المستخدم أية فرصة لتأكيد العمل .

وبما أن مقاطعة لوحة مفاتيح الـ BIOS لا تملك تابعاً من أجل هذا , اذا" هذا البرنامج المعرف من قبل المستخدم سيفي بالغرض .

```

C:\WINDOWS\System32\cmd.exe
NOKEYC - (c) 1992 by Michael Tischer
Keyboard buffer purged when counter reaches 0.
 10  9  8  7  6  5  4  3  2  1
Characters in keyboard buffer :
<None>
D:\>

```

الشكل 6-2

أثناء تنفيذ هذا البرنامج إذا لم يتم إدخال أي حرف سيبقى الـbuffer فارغاً , وبعد الانتهاء من كتابة الأعداد تنازلياً من 10 إلى 1 ستكتب العبارة الموضحة بالشكل " characters in keyboard buffer : <None> " .

أما لو ضغط أي مفتاح أثناء التنفيذ سيخزن هذا المفتاح في الـ buffer ويظهر بعد الانتهاء من كتابة الأعداد السابقة على الشاشة كما هو موضح:

```

C:\WINDOWS\System32\cmd.exe
NOKEYC - (c) 1992 by Michael Tischer
Keyboard buffer purged when counter reaches 0.
 10  9  8  7  6  5  4  3  2  1
Characters in keyboard buffer :
102 (f)
103 (g)
109 (m)
102 (f)
103 (g)
102 (f)
107 (k)
100 (d)
106 (j)
102 (f)
107 (k)
106 (j)
102 (f)
111 (o)
100 (d)
D:\>

```

الشكل 7-2

....\PROGRAMS\C

ملاحظة : البرنامج موجود في المجلد
الملف : NOKEY.C

3 برنامج خدمة مقاطعة لوحة المفاتيح :

إن برامج خدمة مقاطعة لوحة المفاتيح لها العديد من الاستخدامات و التطبيقات مثل :

- عند الحاجة لتغيير حجم الذاكرة الوسيطة لشيفرات لوحة المفاتيح (KB Buffer).
- عند الحاجة لصنع برامج قيادة لوحة المفاتيح خاصة بنا أو بناء برامج مقيمة في الذاكرة .TSR

و بالتالي لتحقيق مثل هذه المتطلبات فإننا نقوم باستبدال برنامج تخديم المقاطعة القياسي الذي ينصبه برنامج الإقلاع الأساسي في الحاسب الـ BIOS عند بدأ إقلاع الحاسب الشخصي حيث نقوم باستبدال برامج خدمة المقاطعة هذه ببرامج من صنعنا لتقوم بتنفيذ أشياء معينة نحن نحددها عند حدوث هذه المقاطعات الموافقة لهذه البرامج.

و كما رأينا سابقا أن لوحة المفاتيح تتعامل مع المقاطعتين 09H (الصلبة) و 16H (البرمجية) . و بالتالي للقيام بكتابة برامج يتم تفعيلها و تنشيطها اعتماد على لوحة المفاتيح يجب أن نتعامل مع هاتين المقاطعتين. و هذا الفصل يحوي مثالين على ذلك.

3.1 الوصول إلى المقاطعة 16H

كما أسلفنا سابقا أنه إذا كنت تود أن تعدل في سير عمل برنامج خدمة مقاطعة ما فإنه عليك الإستيلاء على برنامج خدمة المقاطعة الأصلي و من ثم إحلال برنامجك الخاص مكانه. و هذه الطريقة تستخدم بالنسبة للمقاطعة الـ 16H للقيام بالعديد من الوظائف منها مثلا لاستبدال تنفيذ الوظائف 00H,01H,02H بمقابلاتها الوظائف الحديثة 10H,11H,12H و بالتالي في كل استدعاء للوظيفة 00H يتم عوضا عن تنفيذ الخدمة القديمة القياسية يتم تنفيذ الخدمات الجديدة التي تقدم لنا إمكانية الوصول إلى مفاتيح لوحات المفاتيح الحديثة من النوع MF-II. و التي تؤمن لنا الإستجابة للأزرار إضافية في هذا النوع من لوحات المفاتيح مثل F11, F12.

3.2 مثال :

إن المثال الذي نقدمه هنا هو عبارة عن برنامج صغير بلغة التجميع من نوع TSR أي أنه برنامج مقيم في الذاكرة عندما تقوم بتنفيذه لأول مرة يتم تحميله إلى البرنامج الذاكرة في منطقة معينة منها طبعا و من يبقى مستقرا هنالك, وهذا البرنامج يتم تنشيطه (تنفيذه) عند ورود المقاطعة و مهمة هذه المثال هي إظهار جملة INFORMATIC FACULTY لدى الضغط على تراكب المفاتيح ALT+N.

3.3 برنامج خدمة مقاطعة جديد.

تعتمد فكرة المثال هنا كما تكلمنا سابقا على تنفيذ برنامج مقيم في الذاكرة يتم تنفيذه لدى استدعاء المقاطعة 16H و بالتالي أمامنا مهمة تعديل برنامج خدمة المقاطعة 16H و تعديل مسار عمله لينفذ الأمور التي نحن نرغب بتنفيذها و التي هي هنا عبارة عن ماكرو يقوم بإظهار عبارة ما لدى الضغط على تراكب مفاتيح معين.

يتألف المثال من ثلاث كتل رئيسية و مميزة.

- 1- البرنامج الذي نود أن نضعه في الذاكرة و الذي ينفذ مهمة الماكرو
- 2- القسم الذي يقوم باستبدال برنامج خدمة المقاطعة القديم ببرنامج خدمة مقاطعة جديد و الذي هو في الأعلى.
- 3- القسم الذي يقوم بإلغاء تنصيب برنامج خدمة المقاطعة الجديد و إعادة البرنامج القديم للمقاطعة.

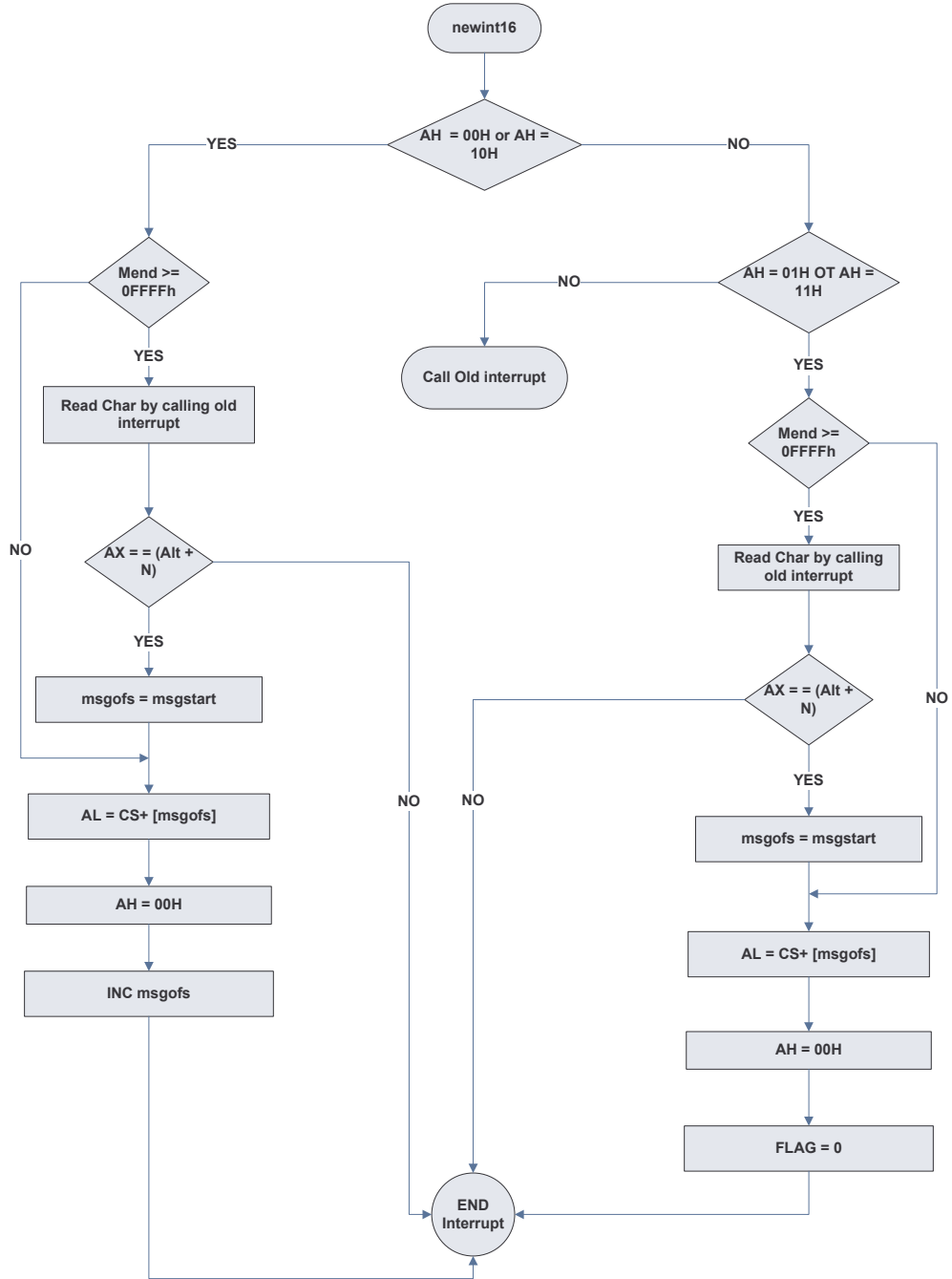
إن ماسبق ذكره مفصل بشكل أكبر في كتب أخرى و ليس هنا مكان لتفصيلها, و لكن ما يهمنا هو القسم الأول و هو برنامج خدمة المقاطعة الجديد.

هذا البرنامج يقوم ببساطة بإعادة نتيجة تنفيذ برنامج خدمة المقاطعة القديم إن لم يكن هنالك ضغط للإختصار الذي نحن ننتظر و هي الـ ALT+N حيث ينفذ برنامج خدمة المقاطعة القديم كما لو أننا لم نحدث أي تغيير. أما في حال ورود اختصار المفاتيح التي انتظرها عندها يتم تنفيذ برنامجي الجديد الذي أضفته, و بالتالي مهمتنا هي هنا ليس إزالة برنامج خدمة المقاطعة القديم كلية و إنما فقط تعديل سير تنفيذها و هذا التعديل يتم بالخطوات التالية :

- 1- تخزين عنوان بداية برنامج خدمة المقاطعة القديم

- 2- تغيير محتوى مؤشر شعاع المقاطعة و التي هي هنا 16H من العنوان القديم للمقاطعة إلى عنوان الموجود فيه برنامجي (هذه هي عملية تنصيب برنامج خدمة المقاطعة
- 3- في كل مرة يلزمي فيه برنامج خدمة المقاطعة القديم فأني استعديه من برنامجي من خلال العنوان الذي خزنته في الخطوة الأولى.

الآن تفاصيل البرنامج الجديد لخدمة المقاطعة مبين بالمخطط التدفقي التالي :



الشكل 1-3

المخطط التدفقي للمقاطعة الجديدة التي سيتم إضافتها لبداية المقاطعة 16H (newint16) ...\\PROGRAMS\\ASM

ملف : MACROKEY.ASM

4 التحكم بالمقاطعة الصلبة للوحة المفاتيح :

عندما نود التعامل مع لوحة المفاتيح عند أدنى مستوى للتعامل فإننا يجب أن نتعامل مع المقاطعة 09H. و مثال على ذلك سوف نقوم بكتابة برنامج خدمة مقاطعة يقوم بالتقاط شيفرة المسح من لوحة المفاتيح مباشرة و عرضها على الشاشة. لكل مفتاح يتم نقره. و هذا البرنامج يعمل بطريقتين فإما أنه يظهر لي شيفرة المسح يقسميها الـ MAKE و الـ BREAK أي لكل محرف سيتم عرض شيفرتين واحدة عند الضغط على الزر و الثاني عند تحرير الزر.

المثال :

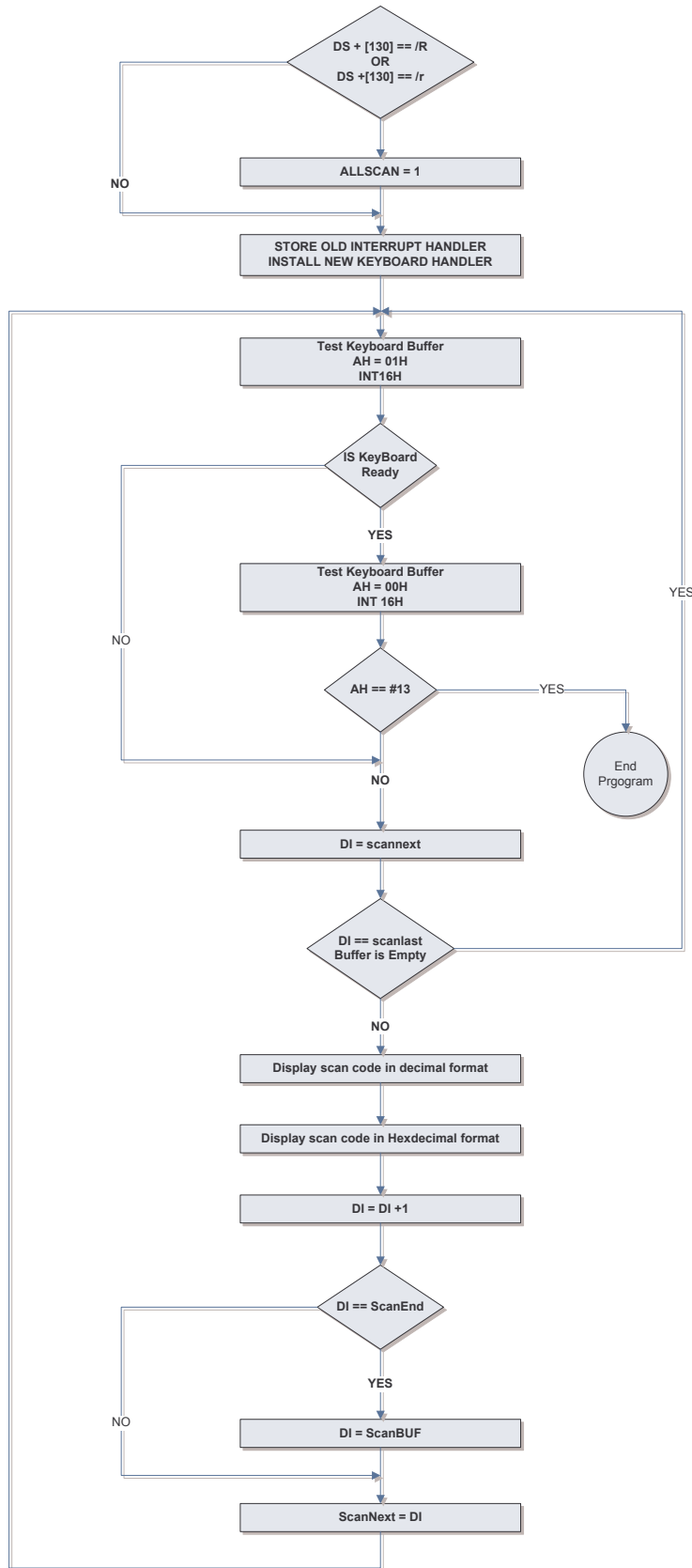
إن المثال الحالي هو عبارة عن برنامج يقوم باستعراض شيفرة المسح لكل زر من أزرار لوحة المفاتيح حيث يمكنك هذا البرنامج من التنصت على لوحة المفاتيح و التقاط الشيفرات القادمة (2 Byte Make & Break Code).

عند تنفيذ هذا البرنامج من خلال محرر الأوامر فإنه يمكننا أن نمرر لهذا البرنامج البارامتر التالي /R او /r و ذلك لكي نخبر البرنامج أننا نريد فقط استعراض شيفرة Make و إلا فإننا نرغب بعرض كل من الـ Make و الـ Break.

إن القسم الرئيسي لبرنامج مثالنا هذا هو برنامج خدمة المقاطعة الجديد الذي سننصبه لتخديم المقاطعة 09H أي مقاطعة لوحة المفاتيح الصلبة. و بالتالي هذا القسم سينفذ في كل مرة يتم ضغط أو تحرير زر على لوحة المفاتيح و بالتالي أصبح مهمته الآن واضحة ما هي :

- 1- يقوم برنامج المقاطعة الجديد باستقبال شيفرتي Make و الـ Break من لوحة المفاتيح على المنفذ 60H.
- 2- تخزين الشيفرات المستقبلية في ذاكرة مؤقتة يتم حجزها من قبل برنامج المقاطعة الجديد و التي يشير إلى بدايتها المتحول الـ SCANBUF و هذه الذاكرة شبيهة تماما بذاكرة لوحة المفاتيح المؤقتة التي يحجزها الـ BIOS, أي أن الذاكرة هي دائرية و يتم التحكم بالتخزين و التفرغ منها من خلال مؤشرين هما الـ SCANNEXT و الـ SCANLAST.
- 3- بعد إجراء الخطوة السابقة و تخزين الشيفرة في الذاكرة يتم مناداة برنامج خدمة المقاطعة 09H القديم و الذي قمنا بتخزين عنوان بدايته في بداية برنامج خدمة المقاطعة الجديد. و بالتالي برنامجنا الجديد لن يحدث أي تأثير ضار بباقي البرامج الأخرى لإننا حافظنا على كل شئ قديم.

أما القسم الآخر من برنامج المثال فهو عبارة عن برنامج عادي يوجد فيه حلقة تكرارية تقوم بقراءة الذاكرة الوسيطة المؤقتة للوحة المفاتيح الخاصة بالـ BIOS من خلال الوظيفتين 01H و 00H. التابعتين للمقاطعة 16H ففي حال تم قراءة المحرف Enter يتم كسر الحلقة و الخروج من البرنامج مباشرة و إلا فإنه يتم قراءة الذاكرة الوسيطة التي قمنا نحن بحجزها لتخديم شيفرات المسح و يعرضها مباشرة على الشاشة بعد أن يقوم بحذفها من الذاكرة. و المخطط التدفقي التالي يوضح ما ذكرناه بشكل مفصل أكثر.



الشكل 1-4

برنامج عرض شيفرة المسح للوحة المفاتيح بعد تخزين المفاتيح المضغوطة

الشكل 2-4

المخطط التدفقي للإجرائية التي ستضاف إلى بداية برنامج خدمة المقاطعة 09H.

....\PROGRAMS\ASM

ملاحظة : البرنامج موجود في المجلد
GETSCAN.ASM : الملف

5 برمجة المتحكم بلوحة المفاتيح :

إن لوحة المفاتيح هي عبارة عن وحدة مستقلة ترتبط مع نظام الحاسب الشخصي و هذه الوحدة المستقلة يتم إدارتها من خلال متحكم صغيري خاص بها كما نعلم أن هذا المتحكم له ذاكرته الخاصة و مسجلاته الخاصة به. و مهمة هذا المتحكم الصغيري هي أخبار نظام الحاسب الشخصي بحدوث حدث معين في لوحة المفاتيح و بالطبع هذا الحدث هو النقر على أحد ازرار لوحة المفاتيح و هذه المهمة تتم من خلال إرسال شيفرة المسح للزر الذي تم الضغط عليه أو تحريره و شيفرة المسح هذه هي التي تحدد موقع الزر المضغوط على لوحة المفاتيح.

إن اتصال لوحة المفاتيح مع نظام الحاسب الشخصي يتم من خلال نظام الإتصال التسلسلي التزامني و باستخدام خطي اتصال ثنائي الاتجاه للإرسال و الاستقبال. حيث يستخدم أحد خطي الارسال و الاستقبال لإرسال المعطيات بت و راء الآخر بشكل تسلسلي بينما الخط الآخر يستخدم كساعة تزامن للإرسال. و في كل عملية ارسال يتم إرسال بايت كامل و الذي هو عبارة عن شيفرة المسح كما نعلم حيث أن هذا البايت يتم إرساله بت بت و يكون البت الأول هو البت الأقل أهمية. و يتم إرفاق هذا البايت ببيت التوقف و بت الفحص و بت البداية.

و بالتالي الـ **Block المرسل في كل عملية إرسال يتألف من bit11 يحوي على بايت معطيات.**

ملاحظة :

بت الفحص يقوم على حاسب عدد البتات في حال كانت زوجية ام لا...

كما يجب أن نعرف أن هنالك فرق بين لوحات المفاتيح الخاصة بحواسب PC XT و التي تستخدم المتحكمات ذات النوع Intel 8048 و بين لوحات الفاتيح الخاصة بحواسب الـ PC AT التي تستخدم المتحكمات 8042 حيث أصبحت الخدمات المتاحة في هذه اللوحات أكثر و كما أن اتصالها مع الحاسب الشخصي أصبح أكثر تعقيدا.

أما من الناحية البرمجية و التي تهمننا نحن كمبرمجين فإن عمليات الإتصال هذه و الإدخال و الإخراج تتم من خلال ذواكر وسيطية و عنواين بوابات محددة و تتألف هذه البوابات من :

- 1- مسجل حالة Status Register.
- 2- ذاكرة إدخال Input Buffer.
- 3- ذاكرة إخراج Output Buffer .

هذه الذواكر تستخدم لنقل ما يلي من المعطيات :

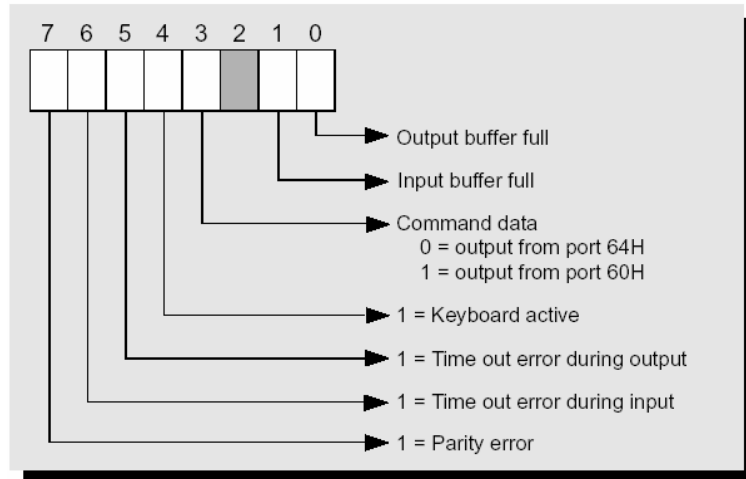
- رموز أزرار لوحة المفاتيح أو شيفرة المسح بشقيها الـ Make & Break.
- المعطيات و المعلومات التي يحتاج نظام الحاسب الشخصي أن يعرفها عن لوحة المفاتيح.

و يمكن التعامل مع هذه البوابات من خلال المنفذ 60H.

يمكن أن نتعامل مع ذاكرة الإدخال Input Buffer من خلال كلا العنوانين التاليين 60H و البوابة 64H. حيث يتم الكتابة في ذاكرة الإدخال هذه من خلال هاتين البوابتين بنفس الطريقة و لكن تستخدم إحداهما لكتابة تعليمات التحكم بمتحكم لوحة المفاتيح 60H و البوابة الأخرى لكتابة المعطيات إلى لوحة المفاتيح على العنوان الـ 64H.

لكن كيف يمكن تمييز لوحة المفاتيح بأن ما نكتبه الآن في الـ Input Buffer هو قادم من العنوان H60 أو 64H ؟ يتم هذا التمييز من خلال بت الثالث في مسجل الحالة المبين في الشكل التالي :

حيث تقوم هذه الأجرائية بفحص المعطيات على البوابة 64H من خلال قراءة محتويات مسجل الحالة بشكل متكرر و في حال كانت البوابة فارغة و مستعدة لاستقبال معطيات جديدة فعندها يتم إرسال بايت التحكم و من ثم يتم فحص وصول البايت إلى لوحة المفاتيح من خلال قراءة مسجل الحالة على البوابة 64H في حال كان الإرسال ناجحاً فيجب أن تكون القيمة المقروءة لمسجل الحالة هي 0FAH و أي قيمة غير هذه فإن ذلك يعني وجود خطأ و بالتالي يجب إنقاص عداد هامش الأخطاء المتاح لإرسال بايت و يتم تكرار العمل مرة أخرى إلى نصل فيها إلى انعدام هامش الأخطاء المتاح للإرسال عندها يتم إلغاء عملية الإرسال و إلا فإن الإرسال يكون قد تم بنجاح.



الشكل 1-5

إن كل من البتتين الأول و الثاني في المسجل يلعبان دوراً مهماً في عملية الاتصال بين الحاسب و لوحة المفاتيح و تحقيق التزامن خلالها كما يلي

البت الأول Bit0: يشير لحالة ذاكرة الإخراج Output Buffer

• Bit0 = 1 : تكون ذاكرة الخرج ممتلئة بالمعلومات و لم تقرأ حتى الآن من قبل الـ Port 60H و في حال تم قراءة هذه الذاكرة فإنه مباشرة سيتم تفسير هذا البت مباشرة.

البت الثاني Bit1: يشير يعمل نفس عمل البت الأول و لكن بالنسبة للـ Input Buffer. حيث أنه في حال كانت قيمة البت = 1 فإنه لن تستطيع أن تكتب أي شيء ضمن هذه الذاكرة.

معدل سرعة الاستجابة للوحة المفاتيح Typematic Rate :

إن من بين العديد من تعليمات التحكم التي يقوم نظام الحاسب الشخصي بإرسالها إلى لوحة المفاتيح يوجد تعليمتين تهما كمبرمجين.

الأمر الأول هو الذي يحدد معدل سرعة تكرار لوحة المفاتيح، أي عدد شيفرات المسح التي يمكن للوحة المفاتيح أن ترسلها إلى نظام الحاسب و ذلك عندما نضغط على الزر و نقيه مضغوطا. و هذا التواتر يبلغ عادة 30 شيفرة مسح في الثانية. و الآن للتحكم بهذا المعدل و لكي لا يكون عشوائيا فإن هنالك إمكانية لضبط ذلك حيث أن التابع الذي يتم تكراره سوف لن ينفذ إلا بعد فاصل زمني محدد يتم وتحيده من خلال المبرمج.

إن معدل التكرار هذا يشفر ثنائيا و الجدول التالي يظهر معدل التكرار و الرقم الثنائي المقابل لها :

Typematic rate codes for the AT keyboard							
Code	RPS*	Code	RPS*	Code	RPS*	Code	RPS*
11111b	2.0	10111b	4.0	01111b	8.0	00111b	16.0
11110b	2.1	10110b	4.3	01110b	8.6	00110b	17.1
11101b	2.3	10101b	4.6	01101b	9.2	00101b	18.5
11100b	2.5	10100b	5.0	01100b	10.0	00100b	20.0
11011b	2.7	10011b	5.5	01011b	10.9	00011b	21.8
11010b	3.0	10010b	6.0	01010b	12.0	00010b	24.0
11001b	3.3	10001b	6.7	01001b	13.3	00001b	26.7
11000b	3.7	10000b	7.5	01000b	15.0	00000b	30.0

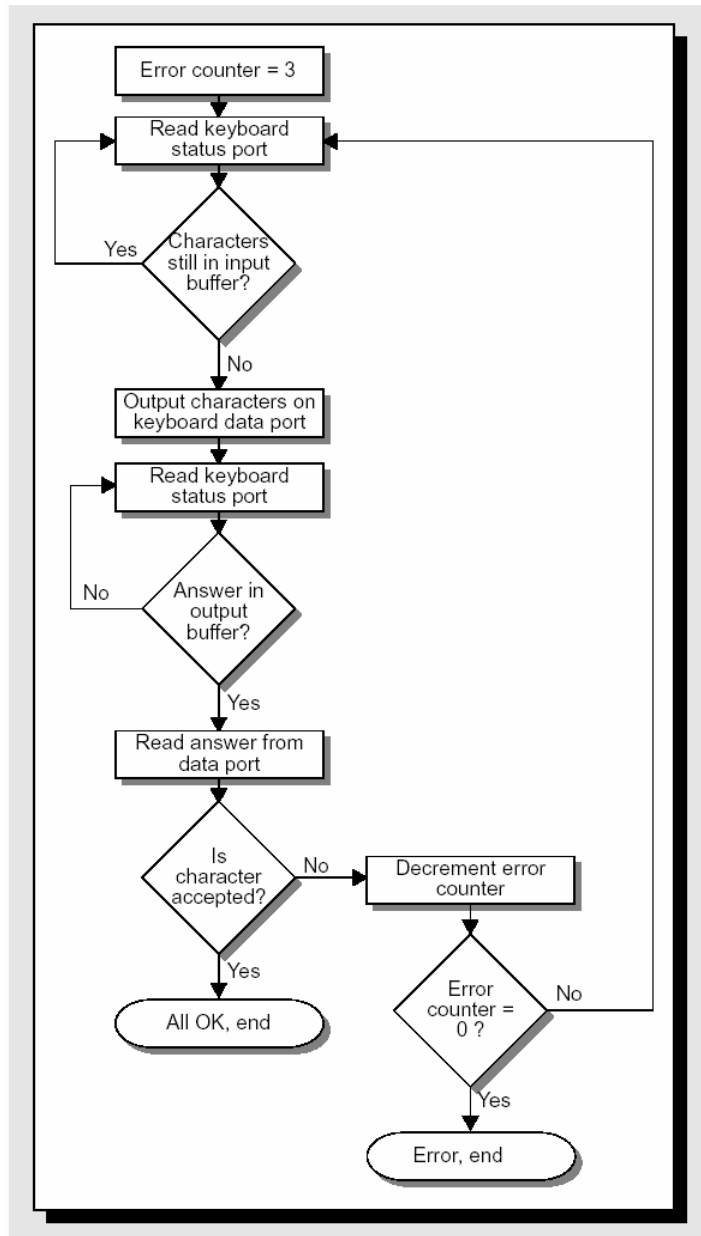
* Repetitions Per Second

جدول 7

للوهلة الأولى قد تعتقد عزيزي القارئ أن الأرقام الثنائية تقابل معدل التكرار بدون قاعدة مسبقة، لا هنالك قاعدة و علاقة رياضية تحدد العلاقة بين معدل التكرار و الرقم الثنائي المقابل له. و هي بفرض ان القيمة الثنائية للبتين الأول و الثاني و الثالث هي A و البت الرابع و الخامس هي B فإن معدل التكرار يكون حسب العلاقة التالية :

$$\text{Repeat Rate} = (8 + A) * 2B * 0.00417 * 1 \text{ second}^{-1}$$

و إن معدل التكرار هذا يتم إرساله إلى لوحة المفاتيح بعد أن نكون قد أرسلنا أمر التحكم المناسب (34H) و هذا الإرسال يتم إلى المنفذ 60H و لكن لن نستطيع أن نرسلها من خلال تعليمة الأسمبلي OUT العادية، يجب أن نستخدم بروتوكول نقل يتضمن قراءة حالة لوحة المفاتيح و يختبر إمكانية النقل و عدم حدوثها. و يجب أن يتم عمل هذا البروتوكول على كلا البائتين المرسلين. و بالتالي علينا أن نكتب إجرائية تقوم بذلك و بنية هذه الإجرائية موضحة بالمخطط التدفقي التالي :



الشكل 2-5

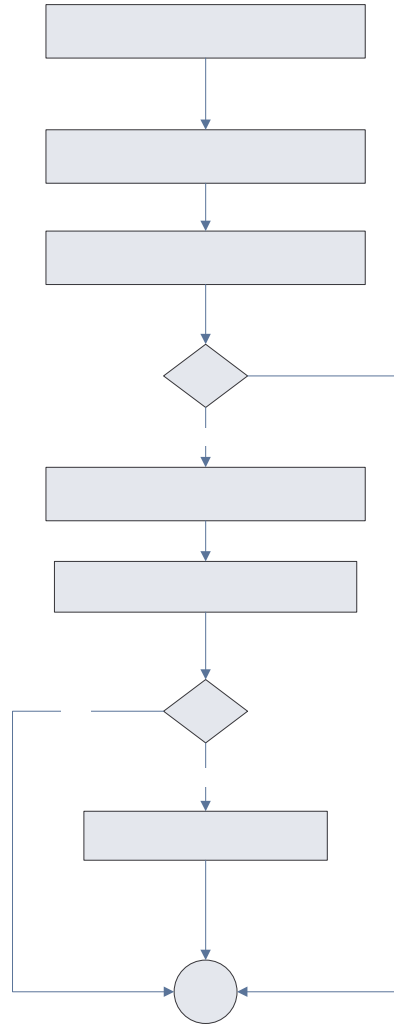
مخطط التدفق لإجرائية ارسال Byte إلى لوحة المفاتيح.

6 أمثلة تطبيقية :

6.1 مثال تغيير معدل التكرار و سرعة لوحة المفاتيح :

كتطبيق على ما ذكرناه سابقا و عن كيفية إرسال أوامر للوحة المفاتيح سوف نقوم من خلال هذا المثال بضبط بارامترات سرعة تكرار المفاتيح على لوحة المفاتيح من خلال إرسال القيم المناسبة حسب وجدنا فيما سبق.

إن الجسم الأساسي للبرنامج هو عبارة عن إجرائية مكتوبة بالـ Assembly و وظيفتها هي القيام بعملية الإرسال للبارامترات للوحة المفاتيح وفق ما سبق و ذكرناه عن إجرائية إرسال بايت للوحة المفاتيح. بالإضافة إلى أننا سنقوم بتمرير قيمة هذه البارامترات من البرنامج المكتوب بلغة الـ C إلى برنامج الأسمبلي هذا و الذي يقوم ببدايته بإرسال كلمة التحكم للوحة المفاتيح و من ثم يرسل البارامتر لها كما هو موضح بالمخطط التدفقي التالي :



TRate0 = Key Parameters Fro

الشكل 1-6

المخطط التدفقي لإجرائية ضبط قيم تكرار لوحة المفاتيح و التأخير

AH = Set_Typm (Keyboard Co

إن قيمة تكرار المفتاح و التأخير يتم إدخالها للبرنامج كبارامترات عند استدعاء البرنامج التنفيذي من محرر الأوامر و هذه القيم يجب أن يكون بين 0 حتى الـ 30 بالنسبة لسرعة التكرار للمفتاح و بين 0 حتى الـ 3 بالنسبة للتأخير.

....\PROGRAMS\C

ملاحظة : البرنامج موجود في المجلد
الملف : TYPMC.C

6.2 مثال التحكم بأضواء الإشارة على لوحة المفاتيح من النوع AT.

الثنائيات الضوئية :

إن الثنائيات الضوئية في لوحة المفاتيح تعتمد على شيفرة أوامر لإضاءتها أو إطفائها (SET/RESET) بما يناسب مع حالة المفاتيح الموافقة لها و شيفرة الأوامر هذه هي الرقم OEDH و بعد أن تنقل هذه التعليمات بشكل صحيح تنتظر لوحة المفاتيح وصول بايت يعكس حالة هذه الثنائيات و كل بت يمثل حالة ثنائي واحد منها و التي تضاء في الحالة (1) .

فمثلا يقوم الـ BIOS عند تفعيل المفتاح CAPSLOCK بجعل قيمة علم داخلي تساوي الواحد و يرسل البايت مع جعل البتات المناسبة في حالة الواحد إلى لوحة المفاتيح لإضاءة الثنائي الموافق .

عادة يقوم المستخدم بتحديد حالة المفاتيح مثل CAPSLOCK بالضغط عليها و لكن بالإمكان تفعيلها بشكل أوتوماتيكي عند بدء برنامج ما و إضاءة الثنائيات لإعلام المستخدم بتفعيلها و بما أن موقع الأعلام (المسؤولة عن تغيير الثنائيات) هو في قطعة متغيرات الـ BIOS و البتات منفصلة و معروفة فيها يصبح من السهل تغييرها .

فيما يلي المخطط الصندوقي لبرنامج بلغة باسكال يقوم لدى تنفيذه بتغيير إضاءة الثنائيات بشكل منتظم .

يستخدم البرنامج إجرائيتين هما SETFLAG,CLRFLAG دخلهما عبارة عن بايت يمثل حالة المفاتيح المطلوبة و يتم اجراء عملية OR مع شعاع الحالة الراهن الذي نحصل عليه بتعريف متحول من الشكل :

BIOSTSBYTE : byte absolute \$0040:\$0017

و باستدعاء المقاطعة 16 يتم تغيير حالة الثنائيات كما هو مطلوب .

....\PROGRAMS\C

ملاحظة : البرنامج موجود في المجلد
الملف : LED.C

6.3 تطبيق تغيير اشكال الأحرف المطبوعة (من اجل اللغة العربية)

لقد قمنا بإنجاز هذا التطبيق بالتعاون مع مجموعة الـ VGA حيث كان مهمة هذا التطبيق هو تغيير أشكال الحروف الانكليزية المخزنة في الذاكرة و من ثم طباعة هذه الاشكال من خلال لوحة المفاتيح و كما تم كتابة هذه البرنامج بالاعتماد التام على مقاطعات الـ BIOS مما يمكننا من كتابة برنامج مستقل تماما عن نظام التشغيل و لذلك قمنا بإنشاء برنامج اقلع بسيط يحتوي على برنامج يقوم بمايلي :

1. تعديل أشكال المحارف المخزن بالذاكرة الحية من خلال المقاطعة 10H و الوظيفة H11 التي تم تناولها بالتفصيل لدى قسم الـ VGA.
2. الدخول في حلقة تقوم بقراءة المحارف المضغوطة على لوحة المفاتيح من ذاكرة لوحة المفاتيح المؤقتة و ذلك باستخدام المقاطعة 0x16H و الوظيفة 0x00H. و التي تم تناولها بالتفصيل هنا في قسم لوحة المفاتيح.

و كما تم استخدام الأدوات التالية لتحقيق برنامج الإقلع المستقل عن نظام التشغيل :

1. المترجم NASM الخاص بترجمة برامج قطاع الإقلع ذات الـ 256KB.
2. البرنامج المحاكي BOCHS الذي يستخدم كآلة وهمية تحاكي عمل الحاسب و ذلك لاختبار برنامج الأقلع على هذا البرنامج حرصا على عدم اعطاب الحاسب الحقيقي.
3. برنامج Partcopy الذي يستخدم لكتابة برنامج الإقلع الناتج عن عملية الترجمة بالمترجم السابق على قطاع الإقلع على القرص المرن مثلا أو على صورة قرص مرن و التي يستخدمها المحاكي السابق للإقلع منها.
4. برنامج المحاكي للمعالج 8086 ليس هنا فائدة كبيرة و إنما فقط للتحقق من آلية سير خورزمية البرنامج اثناء التنفيذ.

إن هذه البرامج المذكورة مرفقة مع المشروع في مجلد Tools.

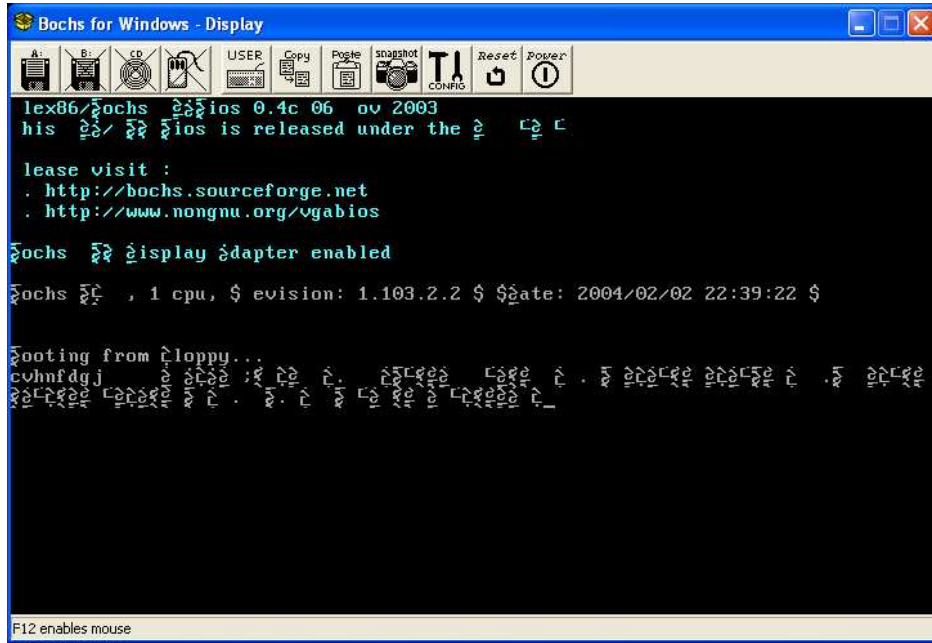
تنفيذ المثال : للقيام بتنفيذ المثال مباشر يجب القيام بالخطوات التالية :

- إذا أردت اختبار المثال على الحاسب الفيزيائي مباشرة قم بالتالي :

1. قم بفتح مجلد الـ /1/arbknk.
2. ضع قرص مرن في السواعة المرنة
3. نفذ من Command Line البرنامج Partcopy بالشكل التالي :
4. اعد اقلع الحاسب من القرص المرن و سينفذ معك المقال إن شاء الله.

- إذا اردت اختبار المثال من خلال المحاكي فيجب القيام بالخطوات التالية :

1. قم بتنصيب المحاكي Bochs من خلال المجلد التالي /tools/bochs/
2. و من ثم شغل الملف a.bxrc و سيقلع البرنامج من خلال المحاكي.



الشكل 2-6

خروج البرنامج على المحاكي

ملاحظة : البرنامج يقوم بتعديل صور المحارف الكبير لشفرة الآسكي.

....\PROGRAMS\ASM

ملاحظة : البرنامج موجود في المجلد

الملف : ARBK.ASM

7 الملاحق

7.1 جدول شيفرات المسح :

Table 10-3. Scan Codes Assigned to Keys on the Standard or AT Keyboard

Key	Scan Code
ESCAPE	1
1 2 3 4 5 6 7 8 9 0 - =	2-13
Backspace	14
Tab	15
Q W E R T Y U I O P []	16-27
ENTER	28
CTRL	29
A S D F G H J K L ; ' `	30-41
SHIFT (left)	42
\	43
Z X C V B N M , . /	44-53
SHIFT (right)	54
Print Screen	55
ALT (left)	56
SPACEBAR	57
CAPS LOCK	58
F1 to F10	59-68
NUM LOCK	69
SCROLL LOCK	70
7 8 9 (Numeric keypad)	71-73
gray -	74
4 5 6 (Numeric keypad)	75-77
gray +	78
1 2 3 (Numeric keypad)	79-81

Key	Scan Code
0 (Numeric keypad)	82
DELETE	83
SYSTEM REQUEST	84

جدول 8

Table 10-4. "Second Codes" for ASCII Code 00h Key Combinations

Key	Scan Code
NUL character	3
SHIFT-TAB	15
ALT-Q,W,E,R,T,Y,U,I,O,P	16-25
ALT-A,S,D,F,G,H,J,K,L	30-38
ALT-Z,X,C,V,B,N,M	44-50
F1 to F10	59-68
HOME	71
UP ARROW	72
PAGE UP	73
LEFT ARROW	75
RIGHT ARROW	77
END	79
DOWN ARROW	80
PAGE DOWN	81
INSERT	82
DELETE	83
SHIFT-F1 to F10	84-93
CTRL-F1 to F10	94-103
ALT-F1 to F10	104-113
CTRL-PRINT SCREEN	114
CTRL-LEFT ARROW	115
CTRL-RIGHT ARROW	116
CTRL-END	117
CTRL-PAGE DOWN	118
CTRL-HOME	119
ALT-1,2,3,4,5,6,7,8,9,0,-,=	120-131

Key	Scan Code
CTRL-PAGE UP	132
F11, F12	133, 134

جدول 9

7.2 الملحق 1. أشكال و مخططات الدراسة :

8	الشكل 11 -
9	الشكل 21 -
13	الشكل 12 -
14	الشكل 22 -
15	الشكل 32 -
16	الشكل 42 -
18	الشكل 52 -
19	الشكل 62 -
20	الشكل 72 -
22	الشكل 13 -
24	الشكل 14 -
25	الشكل 24 -
27	الشكل 15 -
29	الشكل 25 -
30	الشكل 16 -
33	الشكل 26 -

7.3 الملحق 2 جداول الدراسة

10	جدول 1
10	جدول 2
11	جدول 3
12	جدول 4
12	جدول 5
17	جدول 6
28	جدول 7
35	جدول 8
36	جدول 9

8 المراجع المعتمدة :

- **PC Intern**
Copyright 1996 in USA
Abcus.
- كتاب برمجة الحواسيب الشخصية بلغة الأسمبلي
ترجمة الأستاذ حيان السيد
دار شعاع
حلب – سورية
2000
- موقع الخوارزمي لبرمجة أنظمة التشغيل
www.alkhwarmi.com
- موقع للحصول على برامج للتحكم بلوحة المفاتيح على الحاسب الخصي بلغة
باسكال.
<http://www.bsdg.org/SWAG/KEYBOARD/index.html>

تم بعونه تعالى.....