جامعـــــة حلب
كلية الهندسة المعلوماتية
قسم الشبكات – السنة الخامسة
2005-2004

لوحة المفاتيح

بإشراف الدكتور

حسن الصالح

إعداد

أريج حاج محرم

ألاء الحطاب

رنا غبشة

مروان الريحاوي

مادة نظم الزمن الحقيقي

حلب

السبت، 01 كانون الثاني، 2005

الفهرس

بسم الله الرحمن الرحيم

1 مقدمـــــة

تعتبر لوحة المفاتيح الجهاز المحيطي الأساسي المستخدم في التعامل مع الحاسب الشخصي و إدخال البيانات النصية و الرقمية إليه, و انطلاقا 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 الصغري.

كما قدمنا في هذه الدراسة العديد من الأمثلة الشارحة للفقرات النظرية و كما حاولنا التعاون مع مجموعة الـ VGA للقيام بمحاولة تعريب المحارف 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 آملين 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 الـDOS .

إن برامج الـTSR (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 الكومبيوترات من الصنف AT يعالج الاتصال بواسطة شريحة INTEL 8042

و بواسطتها فإن هذه الكومبيوترات قادرة على الاتصال ثنائي الاتجاه بين الـCPU و لوحة المفاتيح علما أن الكومبيوترات الأقدم لا تمتلك هذه الخاصة .

1.2 *تحويل الـ*SCAN CODE

يحول معالج لوحة المفاتيح النبضات الكهربائية المحددة بموقع المفتاح إلى عدد يدعى بالـSCAN 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 سيفرة المسح إلى الكومبيوتر .في الـAT يستقبل المتحكم في لوحة المفاتيح الرمز و هذا النقل يكون بشكل تسلسلي ما دام الكابل الواصل بين الكومبيوتر 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 (الصفر و الواحد).و يتم مزامنة نقل بتات شيفرة المسح المنفصلة وفقا لنبضة الساعة تلك .

إذا تـم ضـغط عـدة مفـاتيح معـا فـإن المعـالج يخزنهـا فـي ذاكـرة داخليـة و تحـوي هـذه الـذاكرة مـساحة فارغـة تكفـي لعـشر ضـربات مفـاتيح و الحقيقـة أن المعطيات تمر إلى المعالج بسرعة أكبر بكثير من سرعة طباعة المستخدم.

1.3 شيفرات العمل و القطع

تتـشكل شـيفرة المـسح أيـضا عنـد تحريـر المفتـاح و هـذا يحـدد للنظـام إذا كـان المفتـاح مـايزال مـضغوطا أو تـم تحريـره للتـو و هـذا مهـم جـدا لأنهـا الطريقـة الوحيـدة التـي يـستطيع فيهـا الكومبيـوتر أن يتـرجم و بـشكل صـحيح الحالـة عنـدما يـتم ضـغط أكثـر مـن مفتـاح فـي وقـت واحـد و بـدون هـذه الخاصـة لـن يكـون بالإمكـان تنفيـذ مهـام خاصـة مثـل طباعـة الأحـرف الكبيـرة أو إعـادة تـشغيل الكومبيوتر بواسطة CTRL+ALT+DEL .

يـستخدم النظـام مـا يـسمى شـيفرات العمـل و القطـع للتمييـز بـين شـيفرات المـسح للمفـاتيح المـضغوطة (العمـل) و المفـاتيح التـي تحـررت (القطـع) الفـرق الوحيـد بينهـا هـو البـت 7 الـذي يكـون بقيمـة 1 فـي شـيفرة القطـع و هـذا يـؤدي إلـى 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 لـSHIFT وهي (36H) ثم شيفرة العمل للحرف A و هي (1EH) للكومبيوتر و طالما لم يتسام النظام شيفرة القطع لـSHIFT يفترض أن المفتاحين مضغوطان 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 الكومبيوتر 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 الـASCII :

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

لا يمرر محرف الـASCII المحول مباشرة إلى التطبيق بل يخزن أولا 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 16 H من الـ ROM-BIOS لها عدة توابع متاحة لهذا الغرض .

1.6 استخدام لغات أخرى في لوحات المفاتيح:

على الرغم من أن الـROOMBIOS يعرف معالج لوحة المفاتيح المستخدم الا أن الـDOS يستطيع أن يبدله ببرنامج آخر و بشكل قياسي يعتمد اللغة الأنكليزية و لاستخدام محارف من لغات أخرى يمكن تنصيبها في ملف الـAUTOEXEC.BAT لاستخدامها.

1.7 لوحات مفاتيح الحواسيب الشخصية:

يوجد ثلاثة أنواع قياسية للوحات مفاتيح الحواسيب الشخصية وهي :

لوحة المفاتيح المنتجة من قبل الـIBM 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 أيضا ولكن هذه الرموز لا تدل على ما يحدث داخل اللوحة .

1.8 لوحات مفاتيح الـXT و PC :

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

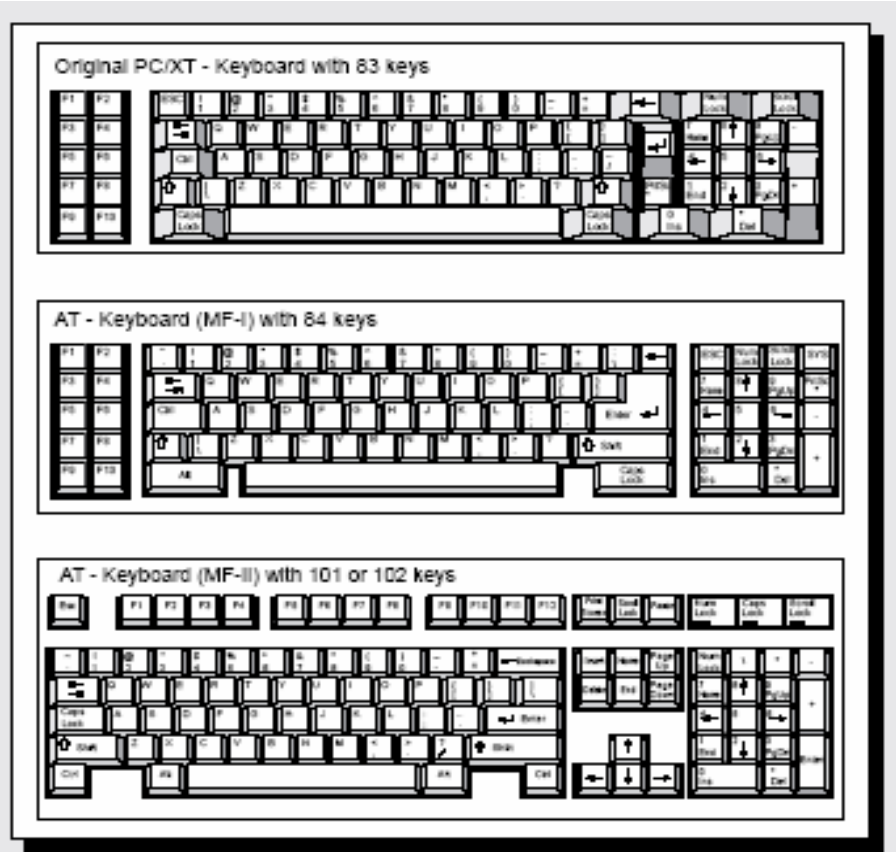
ثم تم انتاج لوحة مفاتيح الـAT التي حلت هذه المشكلة بجعل مفتاح الـEnter و مفاتيح الـ Shift أكبر ومن السهل ايجادها.ولكن مع ذلك كانت تحتوي على مفاتيح صغيرة الحجم مثل الـNum Lock و الـScroll Lock .لأن هذه المفاتيح لا تستخدم بشكل متكرر مثل مفاتيح الـShift .

1.9 لوحة مفاتيح الـ MF-II :

وهي تحتوي على ما يلي :

- مجموعة من مفاتيح أسهم مخصصة منفصلة عن الـkeypad .

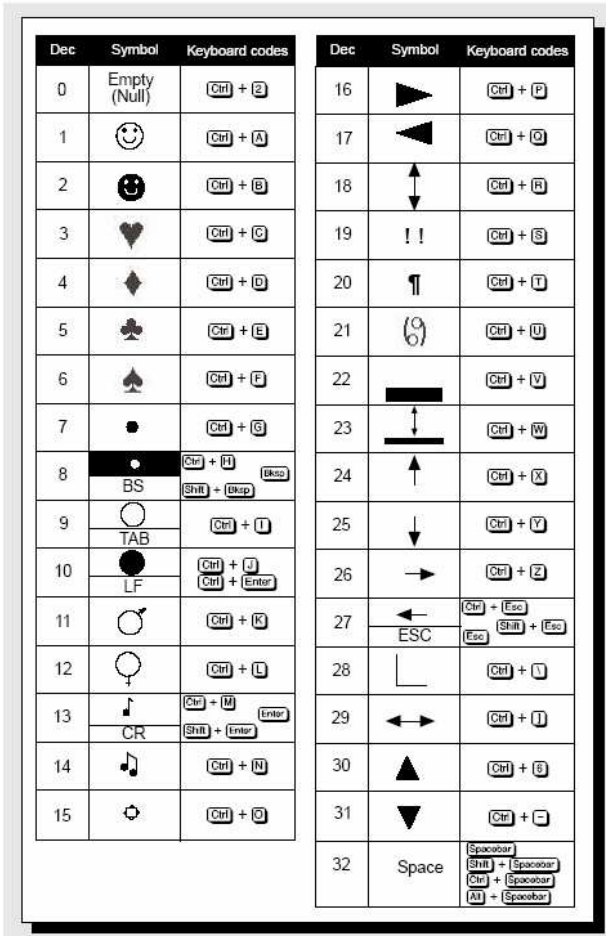
- مفاتيح وظيفية متوضعة في أعلى لوحة المفاتيح.
- مفاتيح الـAlt و التي تتوضع في أسفل اللوحة ويمكن الوصول إليها بسهولة.
- الـLEDs الثلاثة التي تحدد الحالة وهي Num Lock و Scroll Lock وCaps Lock .



الشكل 1-1

1.10 *شيفرات التحكم*

كما نعلم سابقا أنه أي محرف من محارف الـ Asccii يمكن أن يتم ادخاله من خلال لوحة المفاتيح باستخدم مفتاح الـ ALT و مفاتيح لوحة المفاتيح الرقمية كما أنه من الممكن استخدام مفتاح الـ CTRL لتلك الوظيفة أيضا. و 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
0	Empty (Null)	Ctrl + 2
1	☺	Ctrl + A
2	☻	Ctrl + B
3	♥	Ctrl + C
4	♦	Ctrl + D
5	♣	Ctrl + E
6	♠	Ctrl + F
7	•	Ctrl + G
8	BS	Ctrl + H, Bksp, Shift + Bksp
9	TAB	Ctrl + I
10	LF	Ctrl + J, Ctrl + Enter
11	♂	Ctrl + K
12	♀	Ctrl + L
13	CR	Ctrl + M, Enter, Shift + Enter
14	♫	Ctrl + N
15	☼	Ctrl + O
16	►	Ctrl + P
17	◄	Ctrl + Q
18	↕	Ctrl + R
19	‼	Ctrl + S
20	¶	Ctrl + T
21	§	Ctrl + U
22	▬	Ctrl + V
23	↨	Ctrl + W
24	↑	Ctrl + X
25	↓	Ctrl + Y
26	→	Ctrl + Z
27	ESC	Ctrl + Esc, Esc, Shift + Esc
28	∟	Ctrl + \
29	↔	Ctrl +]
30	▲	Ctrl + 6
31	▼	Ctrl + -
32	Space	Spacebar, Shift + Spacebar, Ctrl + Spacebar, Alt + Spacebar

الشكل 1-2

1.11 شيفرة الـASCII :

إذا احتوى المسجل AL على قيمة مختلفة عن الصفر فهي قيمة شيفرة الـASCII و نجد الـCODE SCAN في المسجل AH و ذلك للمفتاح المفعل حاليا مع بعض الاختلافات بالنسبة لمفاتيح التحكم و على سبيل المثال فإن معظم البرامج تترجم الرمز 27 على أنه تعليمة هروب بدلا 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 مضغوطا 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 للـBIOS 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	(2nd keyboard series)
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	(3rd keyboard series)

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	(4th keyboard series)
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	Pg 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	(1st keyboard series)
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)

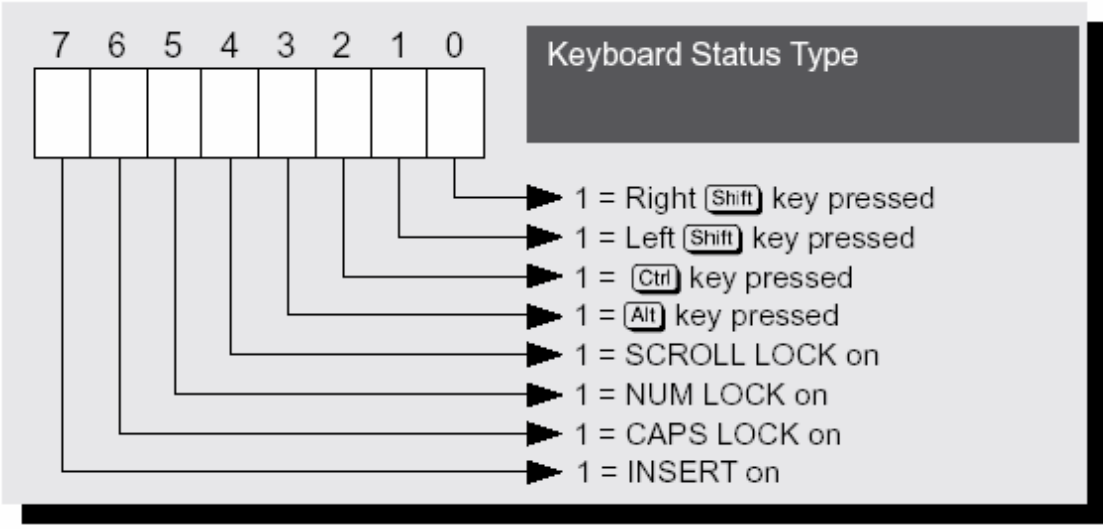
عندما تحدث المقاطعة 16H فإذا كان المحرف موجودا في الـBUFFER فإن يتم حذفه منها و تمريره إلى البرنامج المستدعي أما إذا كانت الـBUFFER فارغة يتم انتظار الإدخال ثم إعادة القيمة إلى البرنامج .

2.2 التابع 01H (قراءة لوحة المفاتيح)

يقوم هذا التابع أيضا بقراءة لوحة المفاتيح مثل التابع السابق و لكن على عكس التابع 00H لا يتم حذف المحرف من الـBUFFER و يتم إعلام البرنامج عن طريق علم التصفير(ZEROFLAG) فإذا كان مساويا الواحد لا يوجد محرف و في الحالة المعاكسة يحوي كل من AL,AH 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 الواحد فإن مفتاح ALT مضغوط علما ان شعاع الحالة هذا مفيد في برامج TSR التي تفحص حالة المفاتيح باستمرار.



الشكل 2-1
(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 :

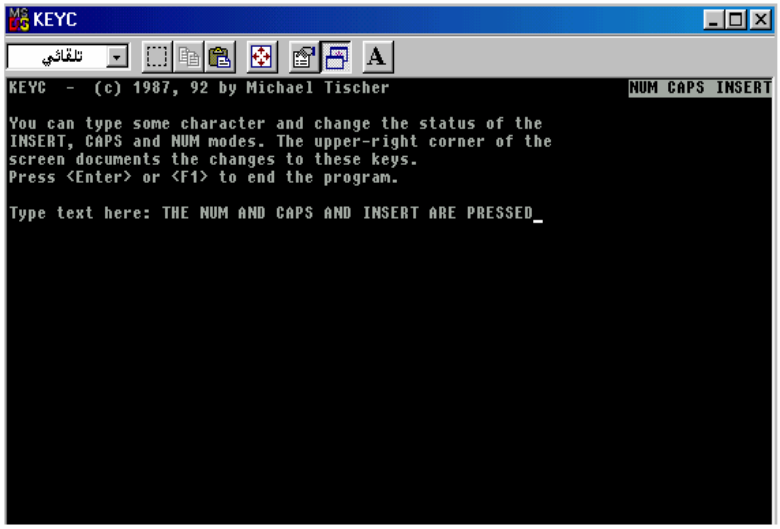
INSERT , CAPSLOCK , NUMLOCK

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 الـBUFFER 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 الى 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

ملاحظة : البرنامج موجود في المجلد ....\PROGRAMS\C

الملف : TYPMC.C

كما انه موجود بلغة الباسكال في المحلد /PROGRAM\PAS

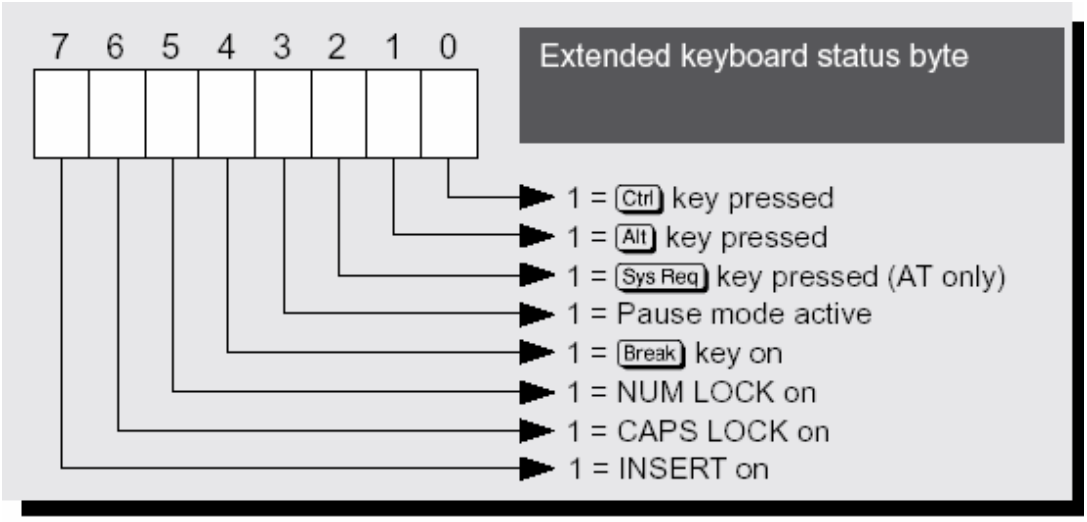
الملف KEYP.PAS

2.5 توابع جديدة في الـBIOS :

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

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

تماثل التوابع 10H , 11H 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 .

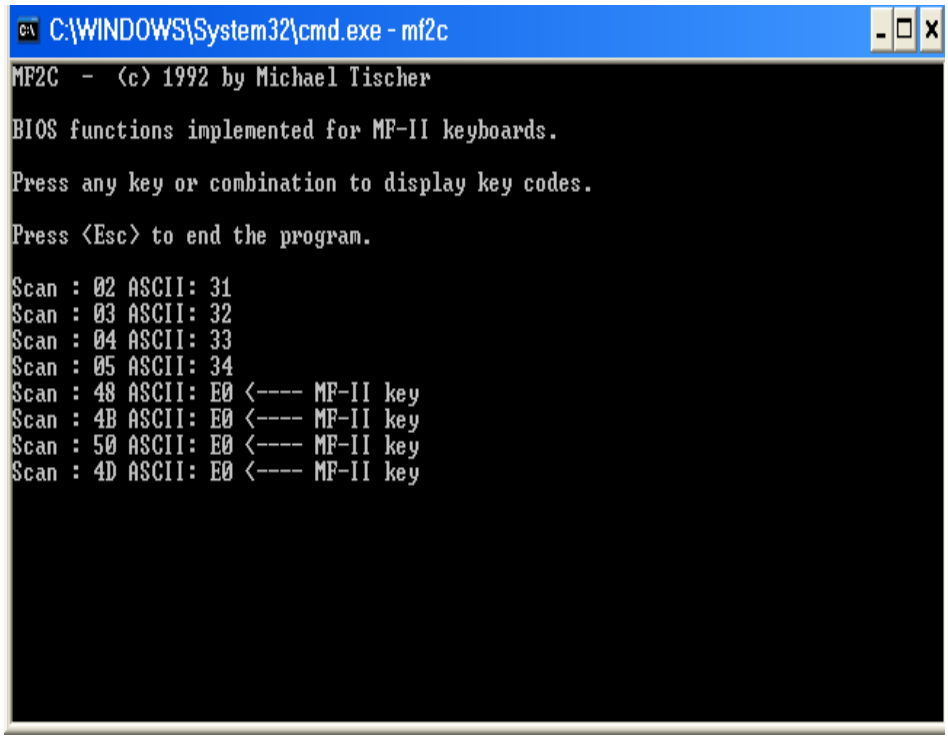


الشكل 2-3

مثال:

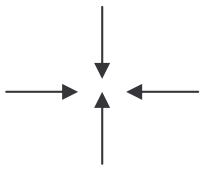
يقوم هذا البرنامج بقراءة لوحة المفاتيح باستخدام تابع الـBIOS (10H) وعرض شيفرة الـASCII والـSCAN المقابلة للحرف المدخل من لوحة المفاتيح , فهو يعرض الشيفرة المقابلة للمفاتيح الخارجية ومجموعة المفاتيح المتوفرة على لوحة المفاتيح MF_II .

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



الشكل 2-4

الشكل السابق يظهر شيفرة الـASCII والـSCAN للمحارف العادية "4 3 2 1 "



والمحارف الموسعة على التوالي.

ملاحظة : البرنامج موجود في المجلد ....\PROGRAMS\C
الملف : MF2C.C

متحولات مقاطعات لوحة مفاتيح الـBIOS :

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

هذه المتحولات مفيدة من أجل برامج الـTSR 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 الـBIOS 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 الـBIOS (16H) .

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

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

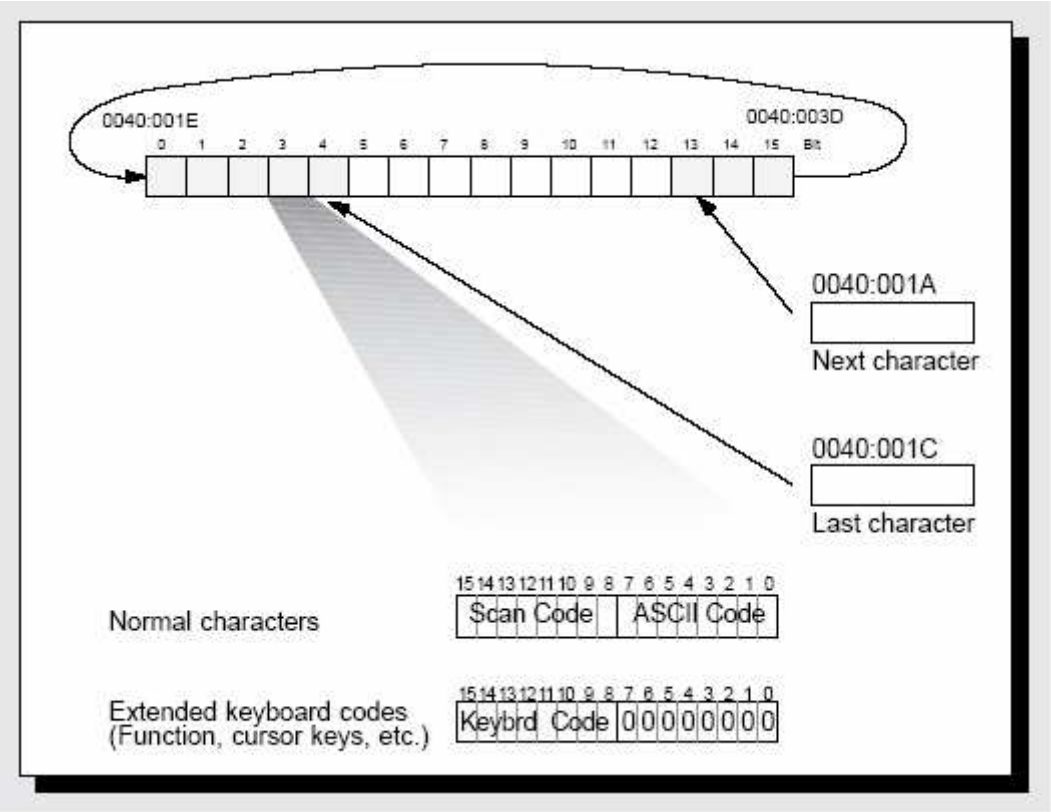
على الرغم من أن هذه الطريقة تعمل بشكل جيد إلا أنها تنجز عملا لا حاجة له بالنسبة للمعالج لأنه في كل مرة يقرأ فيها محرف ما , يجب أن تتحرك كل المحارف الأخرى في الـ buffer , لذلك سيستخدم الـbuffer الحلقي مؤشران: الأول يشير إلى الموقع الذي سيقرأ منه المحرف التالي والثاني يشير إلى المكان الذي سيكتب فيه المحرف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

في البداية يشير كلا من هذان المؤشران إلى بداية المسجل وسيتحركان باتجاه نهايته مع كل قراءة وكتابة لمحرف ما.

عندما يصل كلا المؤشران إلى نهاية الـ buffer ستتم إعادتهما إلى بداية الـbuffer.

ومن حركة المؤشر الدائرية يأتي اسم الـbuffer الحلقي.

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



الشكل 2-5

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

اذا"كل محرف يحتاج بايتين ليخزن في buffer لوحة المفاتيح , البايت الأول للـ ASCII CODE والثاني للـSCAN 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 محرف يحتاج بايتين ليخزن فيه , اذا" يمكن للـbuffer تخزين 16 محرف هذا من أجل محرف الـASCII العادي , أما من أجل كود لوحة المفاتيح الموسعة سيكون الـ ASCII CODE مساويا" للقيمة (0) لأن كود المحرف الفعلي يكون في البايت اللاحق.

موقع buffer لوحة المفاتيح :

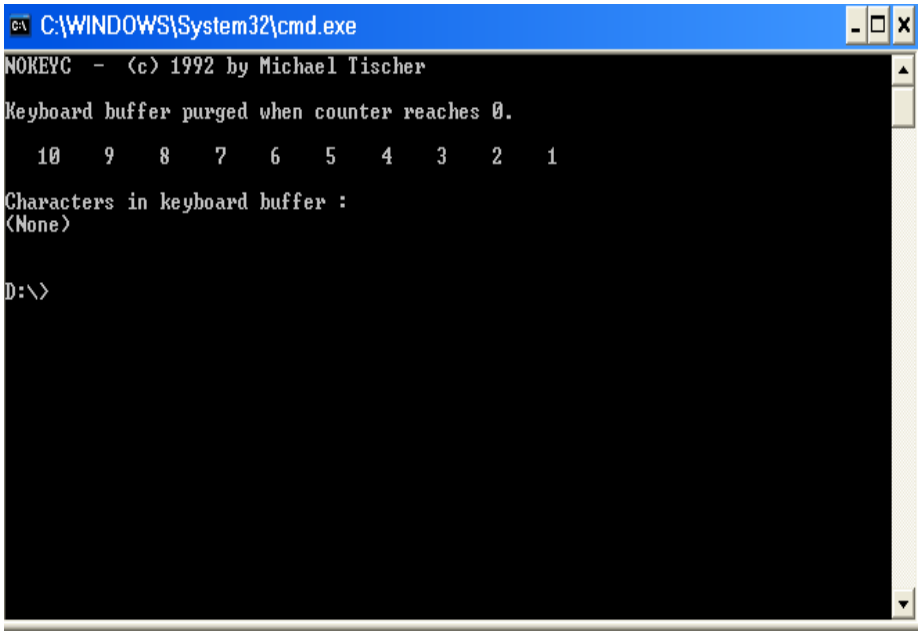
يتوضع buffer لوحة المفاتيح في عنوان الازاحة (1EH) . وكان هذا سبب تصميم (buffers 2) حلقيان , فالمؤشران سيجعلان حجم وموقع buffer لوحة المفاتيح يتغيران عند الحاجة .

مثال: NOKEY.C

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

اذا 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 .

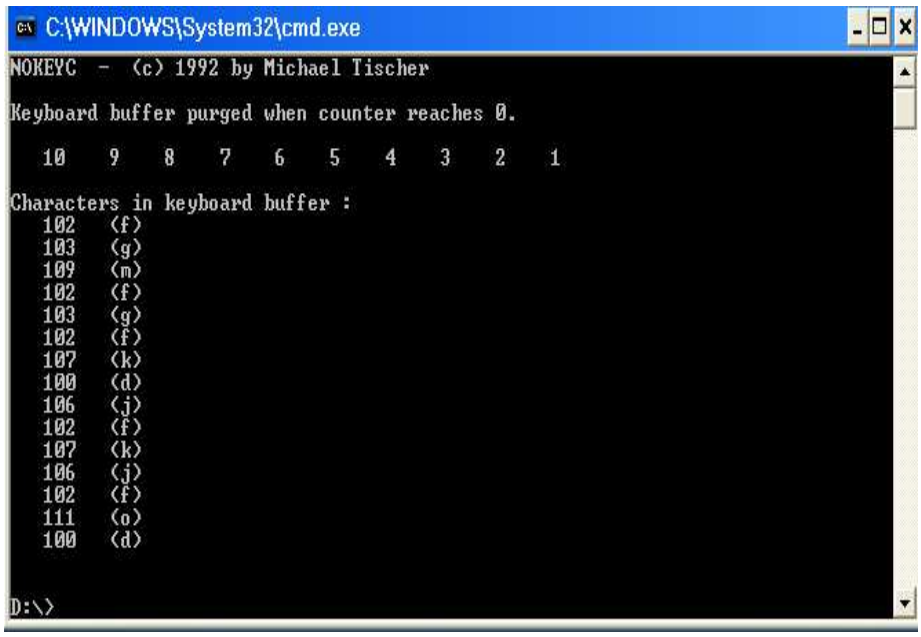
وبما أن مقاطعة لوحة مفاتيح الـBIOS لا تملك تابعا من أجل هذا , اذا" هذا البرنامج المعرف من قبل المستخدم سيفي بالغرض .



الشكل 2-6

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

أمـــا لـــو ضـــغط أي مفتـــاح أثنـــاء التنفيـــذ ســـيخزن هـــذا المفتـــاح فـــي الـــbuffer ويظهر بعد الانتهاء من كتابة الأعداد السابقة على الشاشة كما هو موضح:



الشكل 2-7

ملاحظة : البرنامج موجود في المجلد ....\PROGRAMS\C
الملف : NOKEY.C

3 برنامج خدمة مقاطعة لوحة المفاتيح :

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

- عند الحاجة لتغيير حجم الذاكرة الوسيطية 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 إعتماد عىل 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 ان 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 الوحات 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 باالتالي 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.

يتالف المثال من ثلاث كتل رئيسية و مميزة.

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

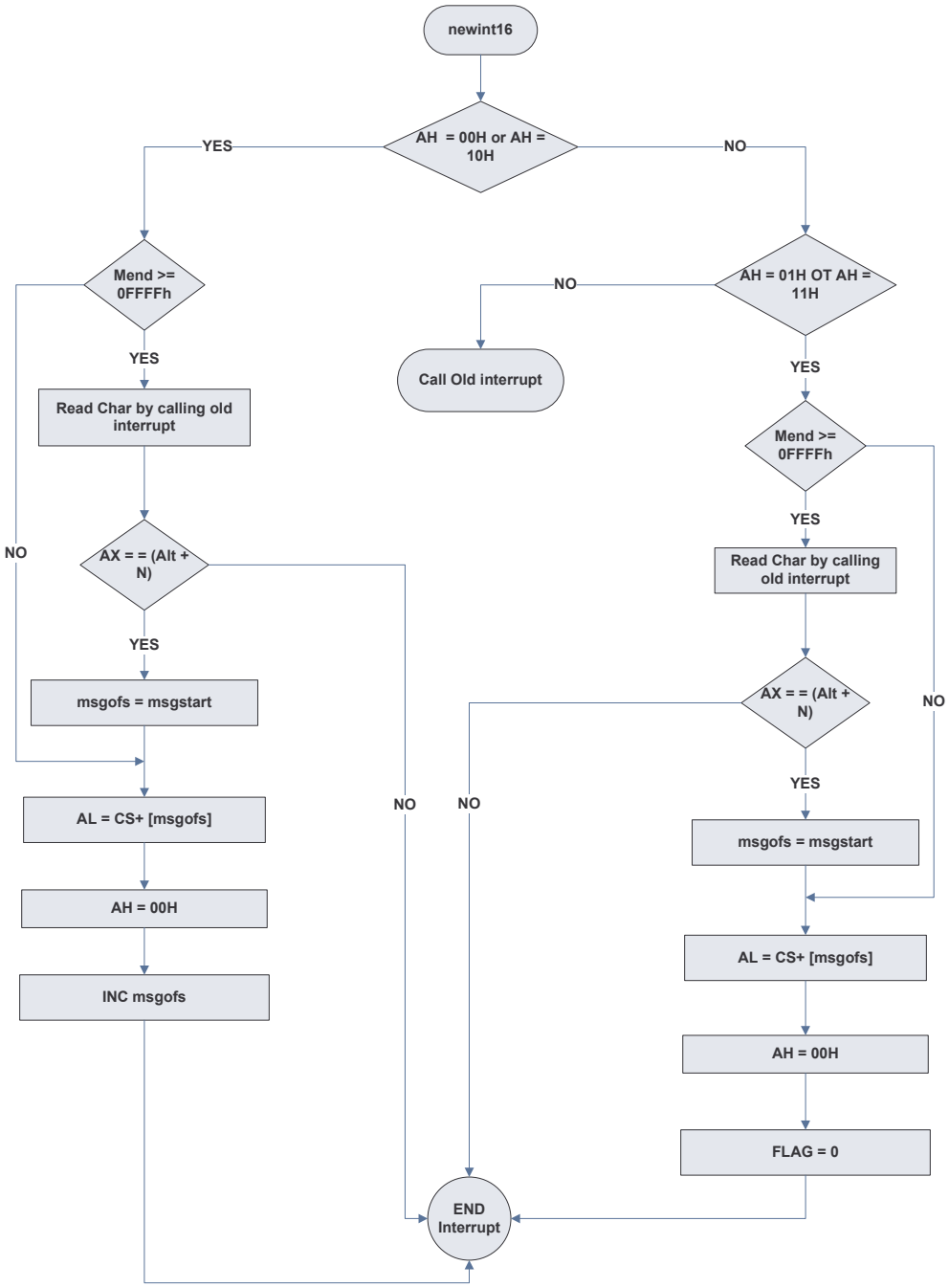
إن ماسبق ذكره مفصل بشكل أكبر في كتب أخرى و ليس هنا مكان لتفصيلها, و لكن ما يهمنا هو القسم الأول و هو برنامج خدمة المقطاعة الجديد.

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 اننا 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 يلزمني 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 3-1

المخطط التدفقي للمقاطعة الجديدة التي سيتم إضافتها لبداية المقاطعة 16H (newint16)

ملاحظة : البرنامج موجود في المجلد ....\PROGRAMS\ASM

الملف : 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 BREAKE أي لكل محرف سيتم عرض شيفرتين واحدة عند الضغط على الزر و الثاني عند تحرير الزر.

المثال :

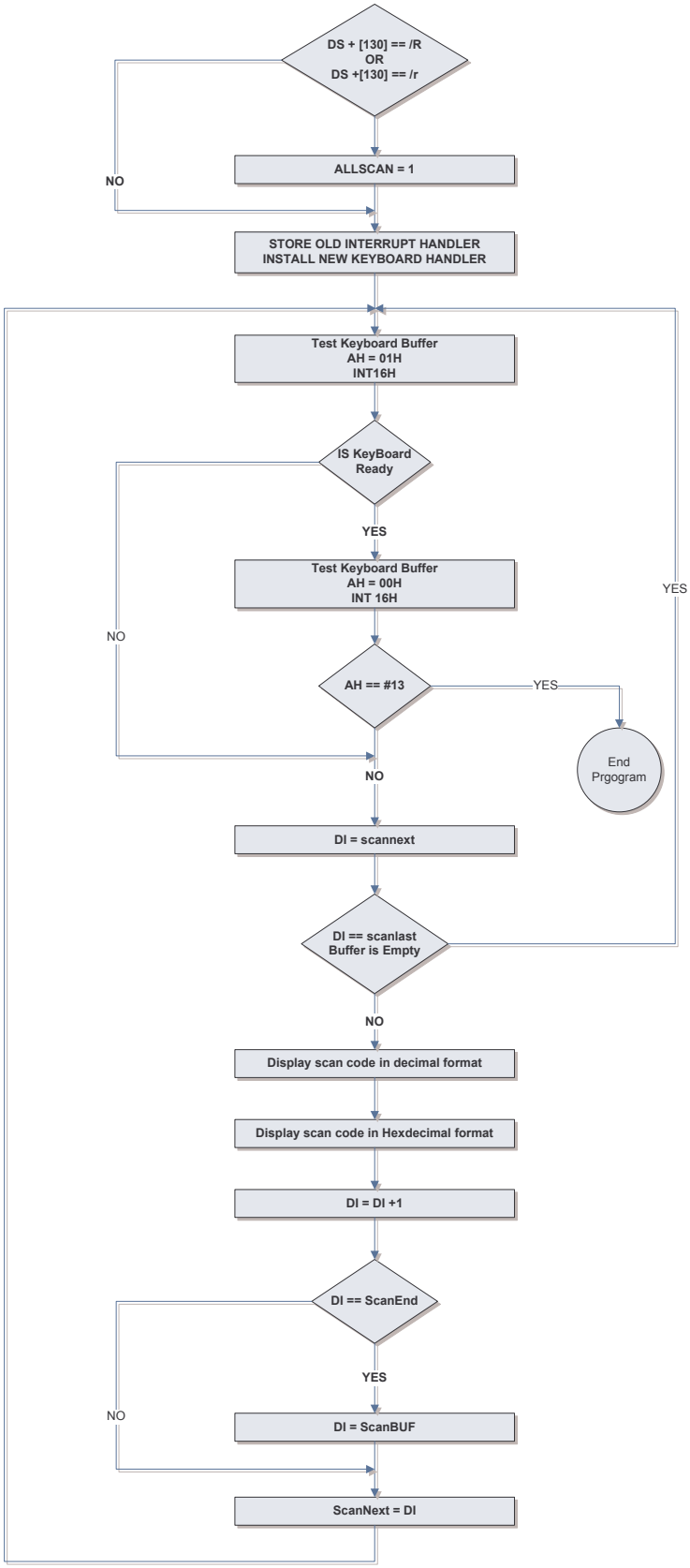
إن المثال الحالي هو عبارة عن برنامج يقوم باستعراض شيفرة المسح لكل زر من أزرار لوحة المفاتيح حيث يمكنك هذا البرنامج من التنصت على لوحة المفاتيح و التقاط الشيفرات القادمة (2 Byte Make & Break Code).

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 شفيرة 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 المستقبلة 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 التفريغ منها من خلال مؤشرين هما الـ SCANNEXT و الـ SCANLAST.
3- بعد إجراء الخطوة السابقة و تخزين الشيفرة في الذاكرة يتم منادة برنامج خدمة المقاطعة 09H القديم و الذي قمنا بتخزين عنون بدايته في بداية برنامج خدمة المقاطعة الجديد. و بالتالي برنامجنا الجديد لن يحدث أي تأثير ضار بباقي البرامج الآخرى لإننا حافظنا على كل شيئ قديم.

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



الشكل 4-1

برنامج عرض شيفرة المسح للوحة المفاتيح بعد تخزين المفاتيح المضغوطة

الشكل 4-2

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

ملاحظة : البرنامج موجود في المجلد ….\PROGRAMS\ASM
الملف : GETSCAN.ASM

5 برمجة المتحكم بلوحة المفاتيح :

إن لوحة المفاتيح هي عبارة عن وحدة مستقلة ترتبط مع نظام الحاسب الشخصي و هذ الوحدة المستقلة يتم إدارتها من خلال متحكم صغري خاص بها كما نعلم أن هذا المتحكم له ذاكرته الخاصة و مسجلاته الخاصة به. و مهمة هذا المتحكم الصغري 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 أ, 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 ثنائيي 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 بتم إرفاق هذا البايت ببت التوقف و بت الفحص و بت البداية.

و بالتالي الـ Block المرسل في كل عملية أرسال 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 االخاصة 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 تهمنا 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 إخراجOutput Buffer .

هذه الذواكر تسخدم لنقل ما يلي من المعطيات :

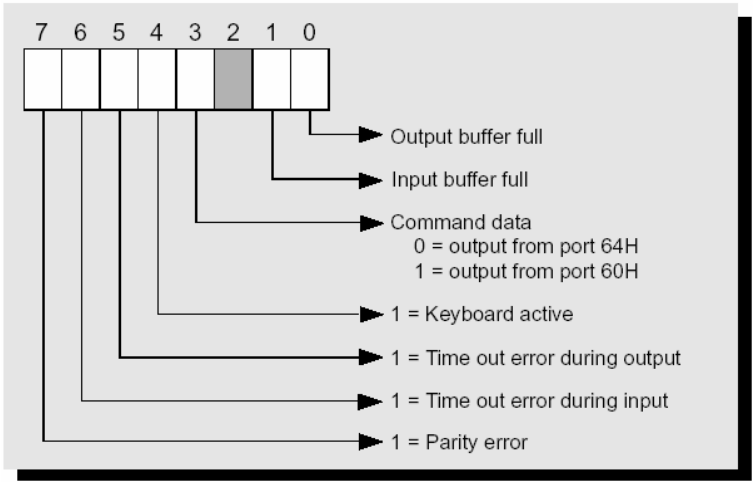
- رموز أرزار لوحة المفاتيح أو شيفرة المسح بشقيها الـ Make & Break.
- المعطيات و المعلومات التي يحتاج نظام الحاسب الشخصي أن يعرفها عن لوحة المفاتيح.

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

يمكن أن نتعامل مع ذاكرة الإدخال Input Buffer من خلال كلا العنواين التاليين 60H والبوابة 64H. حيث يتم الكتابة في ذاكرة الإدخال هذه من خلال هاتين البوابتين بنفس الطريقة و لكن تستخد 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 محتتويات 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 ناجحا فيجب أن تكون القيمة المقرؤة 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 إلأى نصل فيها إلى انعدام هامش الأخطاء المتاح للإرسال عندها يتم إلغاء عملية الإرسال و إلا فإن الإرسال يكون قد تم بنجاح.



الشكل 5-1

إن كل من البتين الأول و الثاني في المسجل يلعبان دورا مهما 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 تصفير 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 شيئ 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 تهمنا كمبرمجين.

الأمر الأول هو الذي يحدد معدل سرعة تكرار للوحة المفاتيح, أي عدد شيفرات المسح التي يمكن للوحة المفاتيح أن ترسلها إلى نظام الحاسب و ذلك عندما نضغط على الزر و نبقيه 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 وتحديده من خلال المبرمج.

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

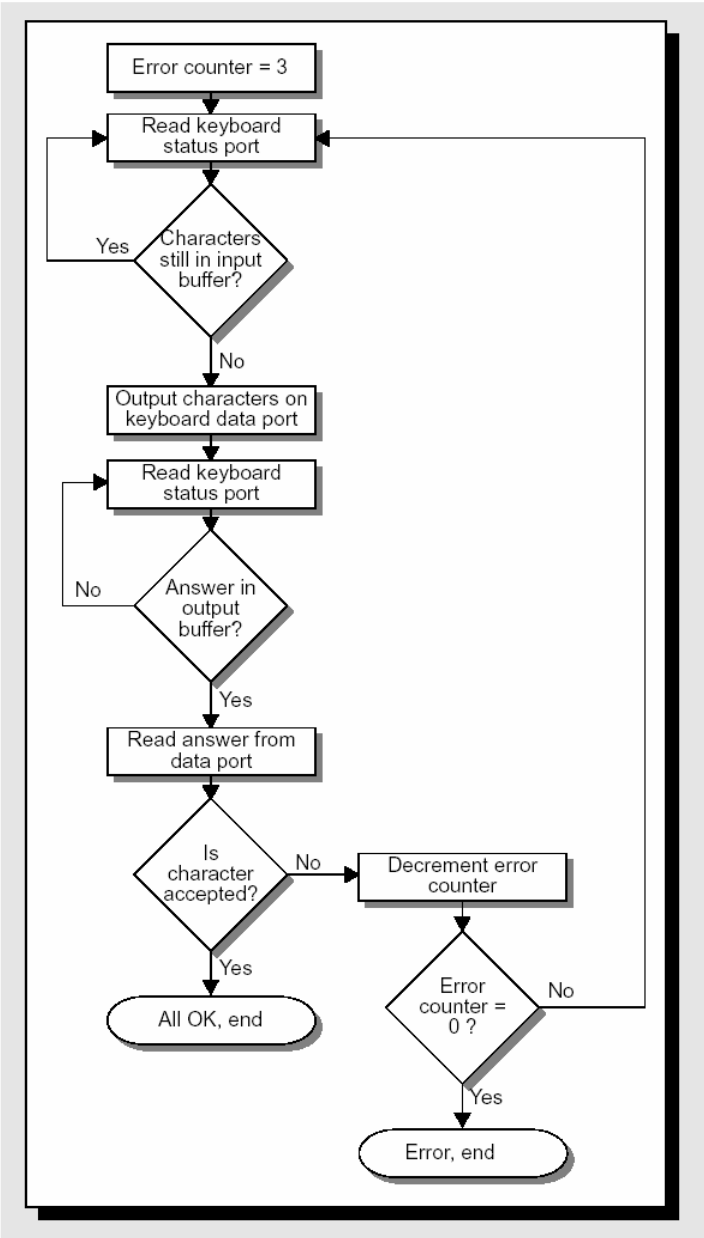
Typematic rate codes for the AT keyboard							
Code	RPS*	Code	RPS*	Code	RPS*	Code	RPS*
11111b	2.0	10111b	4.0	01111b	8.0	00111b	16.0
11110b	2.1	10110b	4.3	01110b	8.6	00110b	17.1
11101b	2.3	10101b	4.6	01101b	9.2	00101b	18.5
11100b	2.5	10100b	5.0	01100b	10.0	00100b	20.0
11011b	2.7	10011b	5.5	01011b	10.9	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 البايتين المرسلين. و بالتالي علينا أن نكتب إجرائية تقوم بذلك و بنية هذه الإجرائية موضحة بالمخطط التدفقي التالي :



الشكل 5-2

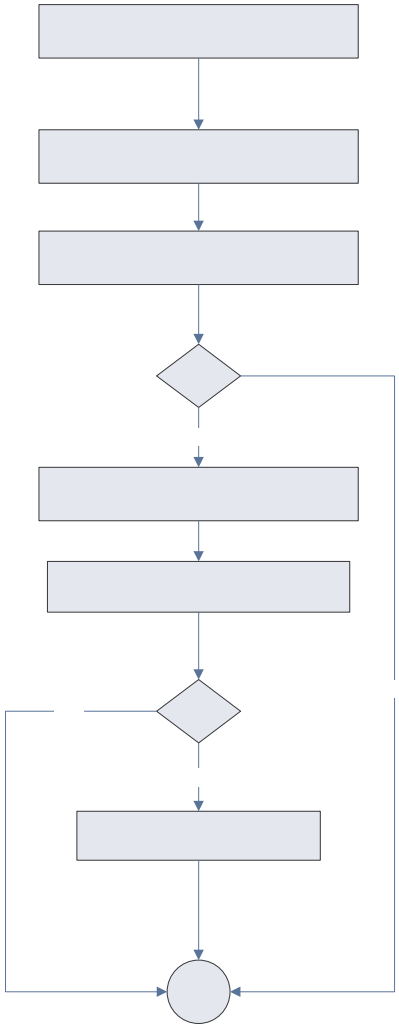
مخطط التدفق لإجرائية ارسال Byte إلى لوحة المفاتيح.

6 أمثلة تطبيقية :

6.1 مثال تغيير معدل التكرار و سرعة لوحة المفاتيح :

كتطبيق على ماذكرناه سابقا و عن كيفية إرسال أوامر للوحة المفاتيح سوف نقوم من خلال هذا المثال بضبط بارامترات سرعة تكرار المفاتيح عل 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 ارسال 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 المفايتح و من ثم يرسل البارامتر لها كما هو موضح بالمخطط التدفقي التالي :



الشكل 6-1

المخطط التدفقي لإجرائية ضبط قيم تكرار لوحة المفاتيح و التأخير

TRate0 = Key Parameters Fro

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 للتاخير.

ملاحظة : البرنامج موجود في المجلد ....\PROGRAMS\C
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 0EDH و بعد أن تنقل هذه التعليمة 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 الـBIOS 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 الـBIOS 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 .

ملاحظة : البرنامج موجود في المجلد ....\PROGRAMS\C
الملف : LED.C

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

لقد قمنا بإنجاز هذا التطبيثق 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 اقلاع 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 الإقلاع المستقل عن نظام التشغيل :

1. المترجم NASM الخاص بترجمة برامج قطاع الإقلاع ذات الـ 256KB.
2. البرنامج المحاكي BOCHS الذي يستخدم كآلة وهمية تحاكي عمل الحاسب و ذلك لاختبار برنامج الأقلاع على هذا البرنامج حرصا على عدم اعطاب الحاسب الحقيقي.
3. برنامج Partcopy الذي يستخدم لكتابة برنامج الإقلاع الناتج عن عملية الترجمة بالمترجم السابق على قطاع الإقلاع على القرص المرن مثلا أو على صورة قرص مرن و التي يستخدمها المحاكي السابق للإقلاع 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.

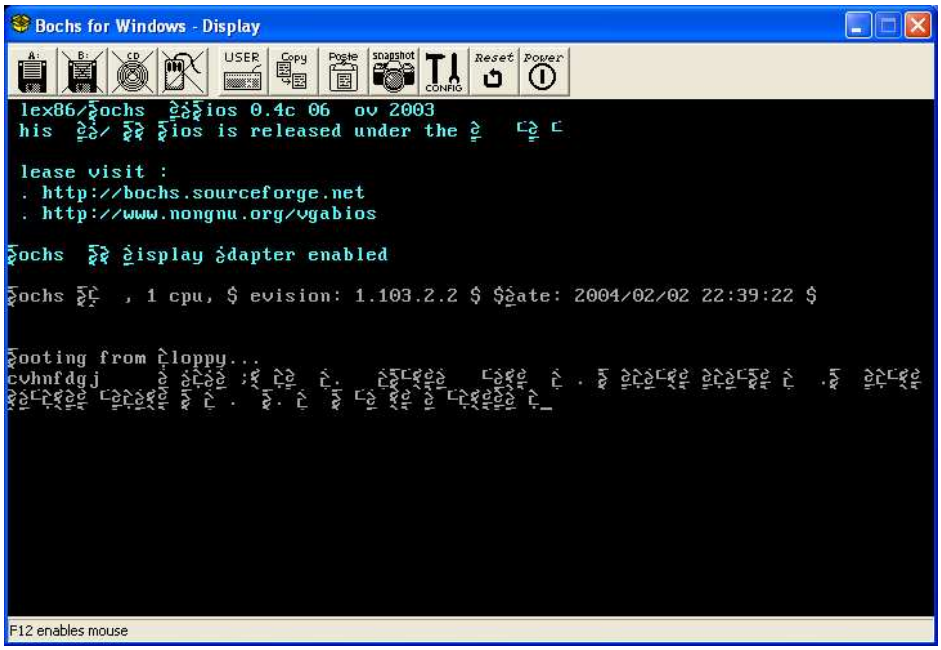
تنفيذ المثال : للقيام بتنفيذ المثال مباشر يجب القيام بالخطوات التالية :

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

1. قم بفتح مجلد الـ arnk/1.
2. ضع قرص مرن في السواقة المرنة
3. نفذ من Command Line البرنامج Partcopy بالشكل التالي :
   …/arbk/1/partcopy arbk 0 200 f0
4. اعد اقلاع 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/boches
2. و من ثم شغل الملف a.bxrc و سيقلع البرنامج من خلال المحاكي.



الشكل 2-6

خرج البرنامج على المحاكي

ملاحظة : البرنامج يقوم بيتعديل صور المحارف الكبير لشفرة الأسكي.

ملاحظة : البرنامج موجود في المجلد ....\PROGRAMS\ASM

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 :

7.3 الملحق 2 جداول الدراسة

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 انظمة 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.....