৪.৮ জেনারেটিং স্কয়ার ওয়েভ ও PWM এর জন্য প্রোগ্রাম উন্নয়ন

(Develop Program for Generating Square Wave and PWM)

প্রোগ্রাম- ১ : টাইমার ব্যবহার করে স্কয়ার ওয়েভ তৈরির 8051 প্রোগ্রাম লেখ।

সমাধান :

নিচে P1.0 পিনে 10 কিলোহার্টসের একটি স্কয়ার ওয়েভ তৈরির প্রোগ্রাম উল্লেখ করা হলো :

```
        MOV     TMOD    ,#02h   ; 8 bit auto-reload mode
        MOV     TH0     ,# – 50 ; – 50 reload value in TH0
        SETB    TR0             ; start timer
LOOP :  JNB     TF0,    LOOP    ; wait for overflow
        CLR     TF0             ; clear timer overflow flag
        CPL     P1.0            ; toggle port bit
        SJMP    Loop            ; repeat
        END
```

প্রোগ্রাম- ২ : টাইমার ব্যবহার করে PWM উৎপন্ন করার 8051 প্রোগ্রাম লেখ।

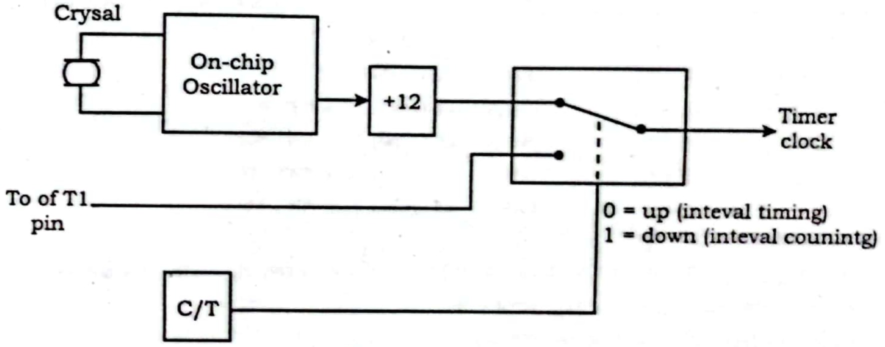
সমাধান :

```
; Timer setup for PWM
CODE :
            PWMPIN EQU P1.0         ; PWM output pin
PWM_SETUP :
            MOV TMOD,# OOH          ; Timer 0 in Mode 0
                                    ; Set pulse width control the value loaded in
                                    R7 is value X as; discussed above.
            SETB EA                 ; Enable Interrupts
            SETB ET0                ; Enable Timer 0 Interrupt
            SETB TR0                ; Start TimerRET â
; Interrupt Service Routine
CODE :
          TIMER_0_INTERRUT
          JB F0, HIGH_DONE      ; If F0 flag is set then we just finished the high
                                  section of the
                                ; cycle so Jump to HIGH_DONESETB F0 Make F0 =
                                  1 to indicate start of high section
LOW_DONE :
          SETB PWMPIN           ; Make PWM output pin HIgh
          MOV TH0, R7           ; Load high byte of timer with R7 (pulse width
          CLR TF0                 control value)
          RETI                  ; Clear the Timer 0 interrupt flag
                                ; Return from Interrupt to where the program came from
HIGH_DONE :
          CLR F0                ; Make F0 = 0 to indicate start of low section
          CLR PWMPIN            ; Make PWM output pin low
          MOV A,#0FFH           ; Move FFH (255) to A
          CLR C                 ; Clear (the carry bit) so it does not affect the
          SUBB A, R7              subtraction
          MOV TH0, A            ; Subtract R7 from A.a = 255 – R7
          CLR TF0               ; So the value loaded into TH0 + R7 = 255
          RETI                  ; Clear the Timer 0 interrupt flag
                                ; Return from Interrupt to where the program came from.
```

৪.৯ ইভেন্ট কাউন্টার হিসেবে টাইমারের ব্যবহার

(Describe the Timer as an Event Counter)

টাইমার শুধুমাত্র সময় পরিমাপ নয় বরং এটি ঘটনা গণনায়ও ব্যবহৃত হয়। যদি Clock source সার্কিট (নিচের চিত্রে) $C/\bar{T} = 1$ হয়, তবে টাইমার ক্লক বাহ্যিক উৎস থেকে হয়। অধিকাংশ ব্যবহারিক ক্ষেত্রে এই বাহ্যিক উৎস কোন ঘটনা সংঘটনে একটি পালস টাইমারকে সরবরাহ করে। এই অবস্থায় টাইমারকে ইভেন্ট কাউন্টার বলে। যেহেতু টাইমার রেজিস্টার প্রত্যেক ঘটনায় ক্রমবৃদ্ধি হয়, তাই এর TL_x ও TH_x এর মান পড়ে সফটওয়্যারে ইভেন্টের সংখ্যা গণনা করা হয়। বাহ্যিক ক্লক উৎস পোর্ট P3 এর বিকল্প ফাংশন থেকে পাওয়া যায়। P3.4 বিটটি টাইমার T0 এর জন্য বাহ্যিক ক্লক ইনপুট এবং P3.5 বিট টাইমার T1 এর বাহ্যিক ক্লক ইনপুট হিসেবে ব্যবহৃত হয়। কাউন্টার প্রয়োগে বাহ্যিক ইনপুট (Tx) এর 1 থেকে 0 পরিবর্তনে (Transition) এ টাইমার রেজিস্টারের ক্রমবৃদ্ধি হয়। অর্থাৎ যখন বাহ্যিক ইনপুট এক সাইকেলে একটি High এবং পরের সাইকেলে Low প্রদর্শন করে, তখন টাইমার রেজিস্টারের ক্রমবৃদ্ধি ঘটে। এই ঘটনার জন্য দুই মেশিন সাইকেল প্রয়োজন, তাই 1 থেকে 0 পরিবর্তনের জন্য 2 মাইক্রোসেকেন্ড সময় লাগে। অপারেশন ফ্রিকুয়েন্সি 12 মে. হার্টস হলে বাহ্যিক ক্লক ফ্রিকুয়েন্সি সর্বোচ্চ 500 কিলোহার্টস।



চিত্র-৪.৭ : ক্লক উৎস

প্রোগ্রাম- ১ : ইভেন্ট কাউন্টার হিসেবে টাইমার ব্যবহার করে ৮০৫১ প্রোগ্রাম লেখ।

সমাধান :

```
MOV       TMOD,   #01100000B                ; COUNTER1, MODE 2, C/T = 1
                                            ; EXTERNAL PULSE
                  MOV         TH1, #0       ; CLEAR TH1
                  SETB        P3.5          ; MAKE T1 INPUT
AGAIN :   SETB    TR1                       ; START THE COUNTER
BACK :    MOV     A, TL1                    ; GET COPY OF COUNT TL1
                  MOV         P2, A         ; DISPLAY IT ON PROT 2
                  JNB         TF1, BACK     ; KEEP DOING IT IF TF = 0
                  CLR         TR1           ; STOP THE COUNTEWR
                  CLR         TF1           ; MAKE TF = 0
                  SJMP        AGAIN         ; KEEP DOING IT
```

অধ্যায়
০৫

8051 মাইক্রোকন্ট্রোলারের ইন্টারাপ্ট

Understand the Interrupt of the 8051 Microcontroller

৫.০ ভূমিকা
(Introduction)

ইন্টারাপ্ট হলো মাইক্রোপ্রসেসরের প্রতি অনুরোধ। কোন নির্দিষ্ট সার্ভিস রুটিন সম্পন্ন করার জন্য ইন্টারাপ্টের মাধ্যমে মাইক্রোপ্রসেসরকে অনুরোধ করা হয়। এই ধরনের সার্ভিস রুটিনকে ইন্টারাপ্ট সার্ভিস রুটিন বলা হয়। ইন্টারাপ্ট কার্যকরী হলে মাইক্রোপ্রসেসর চলমান কাজকে সাময়িকভাবে বন্ধ রেখে ইন্টারাপ্টের জন্য সার্ভিস রুটিন সম্পন্ন করে। বিভিন্ন মাইক্রোপ্রসেসরে বিভিন্নভাবে বিভিন্ন ধরনের ইন্টারাপ্টের সুযোগ থাকে। সাধারণত মাইক্রোপ্রসেসরে ৩ ধরনের ইন্টারাপ্ট থাকে। যথা-

(ক) হার্ডওয়্যার ইন্টারাপ্ট (Hardware Interrupt)
(খ) সফটওয়্যার ইন্টারাপ্ট (Software Interrupt) এবং
(গ) ইন্টার্নাল ইন্টারাপ্ট (Internal Interrupt)

(ক) **হার্ডওয়্যার ইন্টারাপ্ট (Hardware Interrupt) :** মাইক্রোপ্রসেসরের পিনের মাধ্যমে বা সিগন্যালের মাধ্যমে যেসকল ইন্টারাপ্ট সংঘটিত হয়, তাদেরকে হার্ডওয়্যার ইন্টারাপ্ট বলা হয়। যেমন- NMI এবং INTR পিনের মাধ্যমে হার্ডওয়্যার ইন্টারাপ্ট সম্পাদিত হতে পারে।

(খ) **সফটওয়্যার ইন্টারাপ্ট (Software Interrupt) :** ইন্সট্রাকশনের মাধ্যমে যেসকল ইন্টারাপ্ট সংঘটিত হয়, তাদেরকে সফটওয়্যার ইন্টারাপ্ট বলা হয়। INT এবং INTO ইন্সট্রাকশনের মাধ্যমে সফটওয়্যার ইন্টারাপ্ট হতে পারে।

(গ) **ইন্টার্নাল ইন্টারাপ্ট (Internal Interrupt) :** মাইক্রোপ্রসেসরের অভ্যন্তরীণ কোন অপারেশনের ফলে সৃষ্ট ইন্টারাপ্টকে ইন্টার্নাল ইন্টারাপ্ট বলা হয়।

৫.১ 8051 এর ইন্টারাপ্ট এর উৎসের তালিকা
(List the Source of Interrupt of the 8051)

বর্তমান অপারেটিং প্রোগ্রামের বাইরের কোন ইভেন্ট দ্বারা একটি ইন্টারাপ্টের সূত্রপাত হয়। ইন্টারাপ্ট ইভেন্ট বর্তমান প্রোগ্রামে অসময়ে (asynchronously) সংঘটিত হয়, কারণ এটি কখন সংঘটিত হবে তা আগে জানার কোন উপায় নেই। ইন্টারাপ্ট হার্ডওয়্যার ও সফটওয়্যার উভয় ধরনের হতে পারে। একটি প্রোগ্রাম সাবরুটিন ডাকার মতো (Program subroutine call) ইন্টারাপ্ট মূল প্রোগ্রাম নির্বাহে একটি সাময়িক গতি পরিবর্তন করে। 8051 মাইক্রোকন্ট্রোলারে পাঁচটি ইন্টারাপ্ট উৎস থাকে। যেহেতু এখন RESET কেও একটি ইন্টারাপ্ট উৎস হিসেবে বিবেচনা করা হয়, তাই 8051 এর জন্য ছয়টি ইন্টারাপ্ট উৎসের তালিকা নিচে দেওয়া হলো :

No.	Interrupt	Flag
1.	Timer/Counter 1	TF1
2.	External interrupt	TF1
3.	Serial port	RI বা TI
4.	Timer/Counter 0	TF0
5.	System RESET	RST
6.	External interrupt 0	IE0

৫.২ ইন্টারাপ্ট সার্ভিস রুটিন
(Define Interrupt Service Routine (ISR))

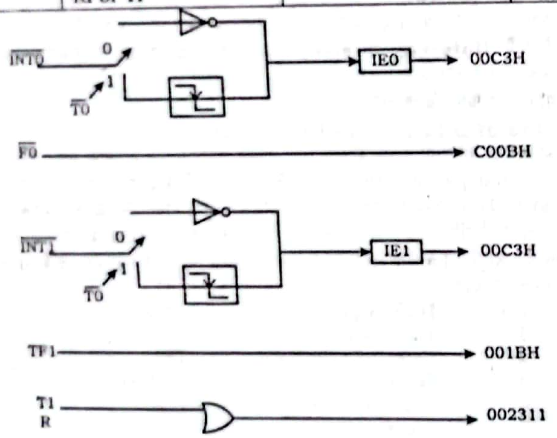
ইন্টারাপ্ট সার্ভিস রুটিন (Interrupt Service Routing) এর সংক্ষিপ্ত নাম হলো ISR। প্রতিটি ইন্টারাপ্টের জন্য একটি করে Interrupt Service Routine (ISR) থাকা আবশ্যক। প্রতিটি ইন্টারাপ্ট এর জন্য মেমোরিতে নির্দিষ্ট লোকেশন থাকে, যা ISR এর অ্যাড্রেস ধারণ করে একে ভেক্টর ঠিকানা বলে। ISR এর অ্যাড্রেসকে ধারণ করার জন্য মেমোরি লোকেশনের গ্রুপকে ইন্টারাপ্ট ভেক্টর টেবিল বলে। চলমান প্রোগ্রামে কোন ইন্টারাপ্ট সংঘটিত হলে যে ছোট প্রোগ্রাম প্রসেসর বা কন্ট্রোলারকে কী করতে হবে তা অবগত করে, তাকে Interrupt Service Routine (ISR) বলে। একে Interrupt handler-ও বলে। ISR ইন্টারাপ্টের প্রত্যুত্তরে সম্পাদিত হয় এবং সাধারণত একটি ডিভাইসের ইনপুট/আউটপুট কাজ সমাধা করে। যখন কোন ইন্টারাপ্ট সংঘটিত হয়, তখন মূল প্রোগ্রাম সম্পাদন সাময়িক স্থগিত হয় এবং প্রোগ্রামকে ISR এ branching করে। ISR সম্পাদিত হয় এবং RETI নির্দেশনার মাধ্যমে সমাপ্ত হয়। ইন্টারাপ্ট সম্পাদন শেষে মূল প্রোগ্রাম (যেখান থেকে ইন্টারাপ্ট শুরু) পুনরায় চলতে থাকে। যখন কোন ইন্টারাপ্ট গৃহীত হয়, তখন PC (Program counter) এ যে মান জমা হয়, তাকে Interrupt vector বলে। এটি ইন্টারাপ্ট উৎসের জন্য ISR এর শুরুর ঠিকানা।

৫.৩ ইন্টারাপ্ট অগ্রগণ্যতা ও ভেক্টর অবস্থান উল্লেখকরণ

(Mention the Interrupt Priority and Vector Locations)

একটি স্বতন্ত্র ইন্টারাপ্ট উৎসকে দুটি অগ্রগণ্যতা স্তরের (priority level) এর একটিতে ধার্য হতে পারে। প্রত্যেক ইন্টারাপ্ট উৎসের জন্য অগ্রগণ্যতা স্তর প্রোগ্রাম করার জন্য একটি বিশেষ রেজিস্টার, Interrupt priority (IP) ব্যবহৃত হয়। উচ্চ অগ্রগণ্যতার ইন্টারাপ্টের একটি ISR রুটিন বিঘ্নিত হতে পারে না। তবে একটি নিম্ন অগ্রগণ্যতার ইন্টারাপ্ট উচ্চ অগ্রগণ্যতার ইন্টারাপ্ট দ্বারা বিঘ্নিত হতে পারে। যদি একই সময়ে ভিন্ন অগ্রগণ্যতার দুটি ইন্টারাপ্ট আসে তবে প্রথমে উচ্চ অগ্রগণ্যতার ইন্টারাপ্টের একটি ISR রুটিন বিঘ্নিত হতে পারে না। তবে একটি নিম্ন অগ্রগণ্যতার ইন্টারাপ্ট উচ্চ অগ্রগণ্যতার ইন্টারাপ্ট দ্বারা বিঘ্নিত হতে পারে। যদি একই সময়ে ভিন্ন অগ্রগণ্যতার দুটি ইন্টারাপ্ট আসে তবে প্রথমে উচ্চ অগ্রগণ্যতার ইন্টারাপ্ট এবং পরে নিম্ন অগ্রগণ্যতার ইন্টারাপ্ট সম্পাদিত হয়। প্রকৃতপক্ষে অগ্রগণ্যতার পরিকল্পনা একটি অভ্যন্তরীণ polling sequence ছাড়া আর কিছু নয়, যাতে 8051 ঐ sequence এ ইন্টারাপ্টকে poll করে। যখন 8051 এ পাওয়ার প্রয়োগ করা হয়, তখন এর ইন্টারাপ্টসমূহের অগ্রগণ্যতা নিচের তালিকা অনুসারে হয় :

ইন্টারাপ্ট উৎস (Interrupt source)	ফ্ল্যাগ (Flag)	অগ্রগণ্যতা স্তর (Priority level)	ভেক্টর ঠিকানা (Vector address)
External interrupt 0	IE0	Highest	0003h
Timer/counter 0	TF0	↓	000Bh
External interrupt 1	IE1	↓	0013h
Timer/counter 1	TF1	↓	001Bh
Serial port	RI or TI	Lowest	0023h



চিত্র-৫.১ : 8051 এর ইন্টারাপ্ট ও ভেক্টর ঠিকানা

৫.৪ ইন্টারাপ্ট এনাবল রেজিস্টার IE বর্ণনা

(Describe Each Bit of the Interrupt Enable (IE) Register)

8051 মাইক্রোকন্ট্রোলারের বিভিন্ন ইন্টারাপ্টের নিয়ন্ত্রণের জন্য IE register ব্যবহার করা হয়। এটি একটি 8 বিটের বিশেষ রেজিস্টার (SFR), যা অভ্যন্তরীণ মেমোরির A8h ঠিকানায় সংরক্ষিত। এটি একটি bit-addressable রেজিস্টার। নিচে এর বিটগুলো সম্পর্কে সংক্ষেপে বর্ণনা করা হলো :

	0	X	0	0	0	0	0	0	Value after reset
IE	EA		ET2	ES	ET1	EX1	ET0	EX0	Bit name
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	

EA – Global interrupt enable/disable :

0- সমস্ত interrupt requests অকার্যকর করে।

1- সমস্ত interrupt requests কার্যকর করে।

ES– Enables or disables serial interrupt :

0- UART system কোন interrupt উৎপন্ন করতে পারে না।

1- UART system interrupt উৎপন্ন করতে পারে।

ET1– Bit enables or disables Timer 1 interrupt :

0- Timer 1 কোন interrupt উৎপন্ন করতে পারে না।

1- Timer interrupt উৎপন্ন করতে পারে।

EX1– Bit enables or disables external 1 interrupt :

0- Pin INT1 logic state এর পরিবর্তনে কোন interrupt উৎপন্ন করতে পারে না।

1- Pin INT 1 logic state এর পরিবর্তনে interrupt উৎপন্ন করতে পারে।

ET0– Bit enables or disables timer 0 interrupt :

0- Timer 0 কোন interrupt উৎপন্ন করতে পারে না।

1- Timer 0 interrupt উৎপন্ন করতে পারে।

EX0– Bit enables or disables external 0 interrupt :

0- Pin INT0 logic state এর পরিবর্তনে কোন interrupt উৎপন্ন করতে পারে না।

1- Pin INT0 logic state এর পরিবর্তনে interrupt উৎপন্ন করতে পারে।

৫.৫ ইন্টারাপ্ট কার্যকর ও অকার্যকর করার প্রক্রিয়া বর্ণনা

(Describe the Procedure of Enabling and Disabling Interrupt)

প্রত্যেক ইন্টারাপ্ট উৎসকে IE (Interrupt enable) রেজিস্টারের মাধ্যমে স্বতন্ত্রভাবে কার্যকর বা অকার্যকর করা যায়। IE রেজিস্টারের EA বিট সেট বা clear করে সমস্ত ইন্টারাপ্ট উৎসকে সর্বজনীনভাবে কার্যকর বা অকার্যকর করা হয়। মাইক্রোকন্ট্রোলার রিসেটে সব ইন্টারাপ্ট অকার্যকর (masked) থাকে, অর্থাৎ এরা অকার্যকরী হলেও মাইক্রোকন্ট্রোলার ইন্টারাপ্ট সম্পাদন করবে না।

প্রোগ্রামে নির্দিষ্টভাবে উল্লেখ করে ইন্টারাপ্টকে কার্যকরী করা হয়। ইন্টারাপ্টকে কার্যকর (masking) বা অকার্যকর (unmasking) করার জন্য নিচের ধাপগুলো অনুসরণ করা হয় :

১। IE রেজিস্টারের D7 বিট (EA) কে অবশ্যই High তে সেট করতে হবে, যাতে রেজিস্টারের অন্যান্য ইন্টারাপ্টগুলো কার্যকর হয়।

২। EA এর মান নির্ধারণ।
যদি EA = 1 হয়, সব ইন্টারাপ্ট কার্যকর হয় এবং IE রেজিস্টারে সংশ্লিষ্ট বিট High থাকে।
যদি EA = 0 হয়, তবে সংশ্লিষ্ট বিট High থাকলেও কোন ইন্টারাপ্টই কার্যকর হবে না।

৩। এবার কোন ইন্টারাপ্টকে কার্যকর করার জন্য ঐ ইন্টারাপ্ট উৎসের Enable bit সেট করতে হয়।

উদাহরণস্বরূপ Timer 1 ইন্টারাপ্টকে কার্যকর করার জন্য কোড নিম্নরূপ :

```
SETB ET1                    ; Enable Timer 1 Interrupt
SETB EA                     ; Set Global enable bit
বা, MOV IE, # 10001000B     ; Set EA and ET1 or IE register.
```

৫.৬ একটি ইন্টারাপ্ট সম্পাদনের ধাপসমূহ উল্লেখকরণ

(Mention the Steps in Executing an Interrupt)

একটি ইন্টারাপ্টের সক্রিয়তার মাইক্রোকন্ট্রোলার নিচের ধাপগুলো অনুসরণ করে :

১। এটি তার চলমান নির্দেশনা সম্পাদনা শেষ করে।

২। এটি প্রোগ্রাম কাউন্টারে রক্ষিত পরবর্তী নির্দেশনার ঠিকানা stack এ জমা করে।

৩। এটি ইন্টারাপ্টের বর্তমান অবস্থা (status) অভ্যন্তরীণভাবে (stack এ নয়) জমা রাখে।

৪। এটি Interrupt vector location থেকে ISR এর ঠিকানা নেয় এবং তাতে জাম্প করে। মাইক্রোকন্ট্রোলার ISR সম্পাদন শুরু করে যতক্ষণ পর্যন্ত না সাবরুটিনের শেষ নির্দেশনা (RETI, Return from interrupt) আসে।

৫। RETI সম্পাদন শেষ হলে মাইক্রোকন্টোলার তার পূর্বের অবস্থানে (যেখান থেকে ইন্টারাপ্ট শুরু হয়) ফিরে আসে। এরপর প্রথমে এটি Stack থেকে প্রথম দুই বাইট প্রোগ্রাম কাউন্টারে POP করে পরবর্তী সম্পাদিত নির্দেশনার ঠিকানা নেয়। তারপর ঐ ঠিকানা থেকে সম্পাদন শুরু করে।