Solutions 5

Question 1:

a) TIM6, being a Basic Timer, is very simple to understand and set up, but it only provides basic functionality. If we wanted more functionality we should use a more advanced timer. {{2}}

b) Both an ISR and a subroutine are a block of code which can be executed and returned from. The differences is that an interrupt (generally) is triggered by hardware at an 'unknown' time, whereas a subroutine is called by software at well defined place in the code. Also, an interrupt automatically pushes a stack frame whereas a subroutine does not. {{2}}

c) For a subroutine, the return address is placed in the link register. For an interrupt, a reserved value is placed into the LR which when moved into the PC informs the CPU that you want to return from the ISR by popping the stack frame. One of the registers in the stack frame is the PC, which is the value of the program counter before the interrupt is triggered. When the stack frame (and hence the PC value is) restored, the CPU goes back to where it was before the interrupt was called.
Hence, both and ISR and subroutine can be returned from by moving the contents of the LR into the PC. {{3}}

d) When the contents of the count register becomes equal to the contents of the auto reload register. {{1}}

e) If the value of the ARR is changed such that it becomes lower than the value in the CNT register, it will take a long time for the update event to be triggered as CNT will have to count up to its maximum value and wrap around before it counts up to the new ARR. This will introduce a long delay in triggering the interrupt which is almost always unwated. By buffering we ensure that the ARR is only actually modified when the CNT value is 0. Hence ensuring that this unwanted behaviour does not occur. {{2}}

Question 2:

a) When a subroutine is called the link register is modified to hold the return address. Hence, for a subroute to be able to return, the contents of the LR must be preserved. If we call a subroutine from within a subroutine, the LR will be modified. Hence, the contents of the LR should first be backed up to the stack before the nested subroutine is called.
Our instruction set only support pushing the LR and popping to the PC. By popping what was in the LR to the PC we effect a return. Hence, we can do `PUSH {LR}` at the start of the subroutine and then `POP {PC}` at the end which will return. This allows calling of other subroutines in between because the LR can safely be modified as it's been backed up to the stack. {{3}}

b) An interrupt driven system prevents having the CPU need to spend its time polling to check whether certain events have occurred which it needs to react to, or spending execution time in long, expensive loops which only take action infrequently. Rather,

interrupts can be used to get the hardware to ‘inform’ the CPU when a certain event occurs, allowing it to react immediately to the event. {{3}}

c) This is a collection of registers which are automatically backed up to the stack when an interrupt happens and automatically restored when the CPU detects that the programmer wants to return from the interrupt handler. This is useful as it ensures that certain parts of the system state are left untouched by the interrupt handler. {{2}}

d) Reset

Question 3:
a) 1/24e6 * (123+1)(456+1) = 2.36 ms {{2}}
b)
delta t = 1s
delta ticks = 2047
time per tick = 1s/2047 = 0,00048852 s

time per tick = 0,00048852 s = (PSC+1)/(8e6)

Hence: PSC+1 = 3908
Hence PSC = 3907
{{2}}