Volume 5, Issue 10, October-2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Running Multi Thread on Arduino Board Wrapped inside Process Control Block using Semaphore

Rahul Jassal
Panjab University Swami Saravanand Gir Regional Centre,
Hoshiarpur, India

Abstract: The paper describes a flavor on parallel programming it is a design of a process control block that is there in generics available on yellow pages. Clone of the context switch help us in understanding the behavior of subroutines inside the micro controller or a times at PC. PCB will handle more than one process and scheduled their status with delays of 1000 millisecond on a PC and same is executed using functions on a suite Arduino Nano 3.0.Obeying the four criteria constraint a process can be scheduled and will imply the same on futuristic processes if they are not executing in their remainder section.

Keywords: PCB, Arduino Board, Critical Section, Process States, Function pointer ().

I. INTRODUCTION

A process can be found in any of four pre-requisite i.e. Entry Section, Remainder Section, Exit Section and Critical Section. While switching from one process to another, these four sections get affected by Boolean values, "True or False" .If $P_B > P_A$ means P_B with more Priority and similar algorithm can be implemented. A process is more than the program code, as it is a program in execution. It also includes the current activity, next activity to be performed and keeping states of processor's register. On the same pattern, process control block contains many pieces of information or manifest associated with a specific process, including Processes State, Program Counter, CPU registers, CPU-scheduling information and Memory management information. It will handle more than one process and schedule their status with delays. In the PCB we came to know about the different phases of the process and can analyze its different states like Entry Section, Remainder Section, Exit Section and Critical Section. We also came to know about how mutual exclusion occurs. So we designed a logic when we switched from one process to another, these four sections get affected by Boolean values "True or False". In first column **of the matrix,** P_A is present in the entry section and all other sections are not in the working state. In second column, P_B is passed from the entry section and is present in the critical section. In the third column, P_C is present in the entry section, at the same time P_A is present in the remainder section which is impossible because it is not possible to present in the remainder section without passing the critical section. In the forth column, time changes from T_0 to T_1, and it comes under the critical section by passing the entry section. In the fifth column, with the passage of time, the status gets affected, as it passes from entry section and critical section, by executing its task completely, it goes to the remainder section.

Process's Status in different Time Intervals...

Time	Critical Section	Entry Section	Exit Section	Remainder Section	Remarks
T0	PA(True)	PA(Passed)	PA(False)	PA(False)	No Comments
T0	PB(False)	PB(True)	PB(False)	PB(False)	No Comments
T0	PC(False)	PC(True)	PC(False)	PC(True)	Not Possible
T1	PA(True)	PA(Passed)	PA(False)	PA(False)	Status Affected
T1	PA(False)	PA(Passed)	PA(False)	PA(True)	Status Affected

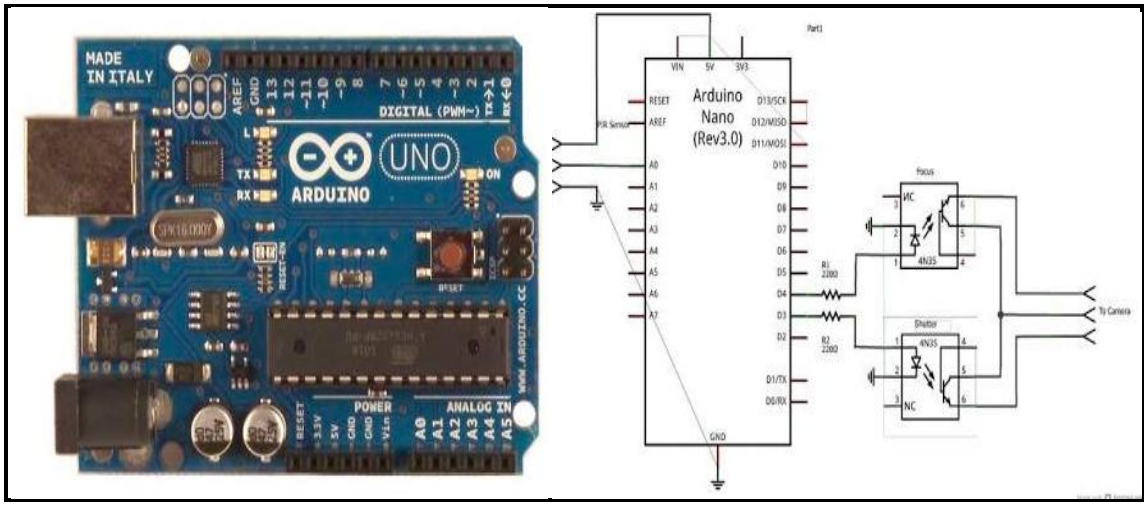
II. IMPLEMENTATION OF ABOVE PROCESS STATUS USING C

```
class process_A
{
char ch1;
int ch;
public:
virtual void display();
friend void show();
void critical();
void exit1();
 };
 void process_A:: display()
  {
```



```
                cout<<"\n\t\t press 1. for Critical Section"<<"\n";
                cout<<"\n\t\t press 2. for Exit Section"<<"\n";
                cout<<"\n\t\t press 3. for Exit from Process_C"<<"\n";
                cout<<"\n\t\t press 4. for Completely Exit from Program Control Block"<<"\n";
                cin>>ch;
        if(ch==1)
        {       critical();         }
        if(ch==2)
        {       exit1();  }
        if(ch==3)
        show();
        if(ch!=1||ch!=2||ch!=3)
        exit(1);
}
 void process_A::critical()
  {
   int ch;          do
        {  cout<<"\t\t\t Welcome To Critical Section"<<"\n";
          cout<<"*********************************************************************************";
          cout<<"\t\t\t\t\t Processes Status"<<"\n";
          cout<<"\t\t\t\t\t **********"<<"\n";
         cout<<"\t\t\t Process A\t Process B\tProcess C"<<"\n";
         cout<<"\t\t\t\t Critical \t Waiting  \t Waiting"<<"\n";
         cout<<"\t\t Continue into Critical Section"<<"\n";
         cout<<"\t\t Press any Key"<<"\n"<<"\n";
         cout<<"\t\t Press 'n'or 'N'to Exit from Critial Section"<<"\n";
         cin>>ch1;
         if(ch1=='n'||ch1=='N')
         break;
        }
        while(1);
                cout<<"\n\t press 1. for Critical Section"<<"\n";
                cout<<"\n\t press 2. for Exit Section"<<"\n";
                cin>>ch;
                        if(ch==1)
                        critical();
                        if(ch==2)
                        exit1();    }
```

III. SUPPORT OF MULTI THREADING ON ARDUINO NANO 3.0



Probably with availability of flavors of parallel programming in market, one must have one or more CPUs that have multiple processing cores. Some operating systems have a utility that visualize processor usage in real time. It is similar to using an android version 4.4.2 with model number HTC 526 G plus. The machine also provides you the sketch of memory usage or sort of activity monitor. Arduino supports Multi threading and has an advantage, it's has the possibility to read multiple captors, blink led and start a motor at virtually the same time. This is done by cycling rapidly (faster as the Arduino can) witting all the functions.


```
typedef void (*FunctionPointer) ();
// Declare looping actions function names, declared lower.
FunctionPointer Xdelegate[] = {loopActionA,loopActionB,loopActionC};
// Define actions status flags. Set to 1 to auto execute a start.
int XdelegateFlags[] = {0,0,0};
int XdelegateCount = sizeof(Xdelegate);
void xActionTrigger(int id=0, int action=0) {
  // The id represent it's position in the flags array.
  // Action 1 = executed, 0 not.
  XdelegateFlags[id] = action;
}
// LOOPING FUNCTIONS
void loopActionA() {
      // Do something...
}
void loopActionB() {
            // Do something...
}
void loopActionC() {
            // Do something...
}
void xDoActions() {
            // Execute all looped function.
            for(unsigned int j=0; j < XdelegateCount; j++) {
                        if( XdelegateFlags[j] == 1 ) { // Execute the action if.
                                    Xdelegate[j](); // Call the related loop action.
                        }           }           }
void setup() {
      xActionTrigger(0,1); // First action.
      xActionTrigger(2,1); // Third action.
}
void loop() {
            xDoActions();
}
```

IV. CONCLUDING REMARKS

This is to be concluded after implementing the two threads one on pc and one on Arduino Board, that more than one task can be performed at the same time. However actual parallel processing is not possible in commonly available computers, but they can simulate such an effect by high speed switching of small amount of CPU time to each of the process that need to be executed. We end up with notes like that can we access two critical sections at the same time or how many times can you switch from one process to another. Is there a way out for switching in the sections of a single process. One can make use of multi threading on such controllers and may opt for IoT in future.

REFERENCES

[1] Multitasking/multithreading in C on Arduino, **http://playwithmyled.com/2009/10/multitasking-multithreading-in-c-on-arduino/, 2009**

[2] Multithreaded C Program on the Raspberry PI, **http://stackoverflow.com/questions/18423885/multithreaded-c-program-on-the-raspberry-pi, 2015**

[3] Process (computing),**https://en.wikipedia.org/wiki/Process_(computing), 2015**

AUTHORS PROFILE

Rahul Jassal is working as Assistant Professor in Department of Computer Science & Application, Panjab University Regional Centre, Hoshiarpur, India. He received his Master Degree in year 2007 and qualifies the lectureship examination for subject "Computer Science & Application in the same year. He area of interest lies with algorithms designs & analysis.

