```c
// filename *************Fifo.c********
// Two implementations of a FIFO
// Jonathan W. Valvano 10/1/07

//  This example accompanies the books
//   "Embedded Microcomputer Systems: Real Time Interfacing",
//        Thompson, copyright (c) 2006,
//   "Introduction to Embedded Microcomputer Systems:
//    Motorola 6811 and 6812 Simulation", Brooks-Cole, copyright (c) 2002

// Copyright 2007 by Jonathan W. Valvano, valvano@mail.utexas.edu
//    You may use, edit, run or distribute this file
//    as long as the above copyright notice remains

#include "Fifo.h"

/* Number of characters in the Fifo
     the FIFO is full when it has FifoSize-1 characters */
unsigned short static volatile *PutPt;    /* Pointer of where to put next */
unsigned short static volatile *GetPt;    /* Pointer of where to get next */
                  /* FIFO is empty if PutPt=GetPt */
                  /* FIFO is full  if PutPt+1=GetPt */
unsigned short static Fifo[FIFOSIZE];     /* The statically allocated fifo data */

/*-----------------------Fifo_Init----------------------------
  Initialize fifo to be empty
  Inputs: none
  Outputs: none */
void Fifo_Init(void){
  unsigned char SaveCCR;
  asm  tpa
  asm  staa SaveCCR
  asm  sei          // make atomic
  PutPt=GetPt=&Fifo[0];   // Empty when PutPt=GetPt
  asm  ldaa SaveCCR
  asm  tap          // end critical section
}

/*-----------------------Fifo_Put----------------------------
  Enter one character into the fifo
  Inputs: 8-bit data
  Outputs: true if data is properly saved,
           false if data not saved because it was previously full*/
int Fifo_Put(unsigned short data){
unsigned short volatile *tempPt;
  tempPt = PutPt;
  *(tempPt) = data;      // try to Put data into fifo
  tempPt++;
  if(tempPt == &Fifo[FIFOSIZE]){ // need to wrap?
    tempPt = &Fifo[0];
  }
  if(tempPt == GetPt){
    return(0);        // Failed, fifo was previously full
  }
  else{
    PutPt = tempPt;  // Success, so update pointer
    return(1);
  }
}

/*-----------------------Fifo_Get----------------------------
  Remove one character from the fifo
  Inputs: pointer to place to return 8-bit data
  Outputs: true if data is valid,
           false if fifo was empty at the time of the call*/
int Fifo_Get(unsigned short *datapt){
  if(PutPt == GetPt){
    return(0);     // Empty if PutPt=GetPt
  }
  else{
    *datapt = *(GetPt);  // return by reference
    GetPt++;             // removes data from fifo
    if(GetPt == &Fifo[FIFOSIZE]){
      GetPt = &Fifo[0];  // wrap
    }
    return(1);
  }
}
```

```c

/*-----------------------Fifo_Status----------------------------
  Check the status of the fifo
  Inputs: none
  Outputs: true if there is any data in the fifo */
int Fifo_Status(void){
  return (PutPt != GetPt);
}
```