Received November 13, 2020, accepted November 25, 2020, date of publication November 30, 2020, date of current version December 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3041373

Optimising SD Saving Events to Maximise Battery Lifetime for Arduino^{TM}/Atmega328P Data Loggers

LUKE J. BRADLEY AND NICK G. WRIGHT, (Member, IEEE)
School of Engineering, Newcastle University, NE1 7RU Newcastle upon Tyne, U.K.

Corresponding author: Luke J. Bradley (luke.bradley@newcastle.ac.uk)

This work was supported by the U.K. Engineering and Physical Sciences Research Council under Grant EP/R511584/1.

ABSTRACT In many areas of science, Arduino based data loggers have become common enabling instruments because of their low cost and ease of use. However, battery life is commonly the limiting factor - particularly in respect of writing data to embedded SD cards. In this paper, various methods by which to optimise an SD card based data logger using an Arduino UNO, Atmega328P at 5 V at 16 MHz and the Atmega328P at 3.3 V at 8 MHz is explored. With the bare Atmega328P chip in sleep mode, the lifetime of a 2400 mAH battery can theoretically exceed 10 years, although this is reduced to only 3 months following the introduction of an SD card. The exact power consumption of an Arduino/SD card during saving events is analysed for the first time and is found to take up to 200 ms with current spikes up to 80 mA for every initialisation and saving event dramatically increasing the average current consumption of fast data loggers. Through the use of a power control MOSFET with proper initialisation and timing of SD saving events, it is found that the Atmega328P can be set up to measure data once every two seconds whilst also ensuring a battery lifetime of one year. With the novel techniques presented here, a new method for maximising the lifetime of Atmega328P microcontroller circuits for environmental data logging applications has been achieved; allowing researchers to record data using a cheap and reproducible system.

INDEX TERMS Data acquisition, data handling, lifetime estimation, low power electronics, microcontrollers.

I. INTRODUCTION

As research progresses, the standard method for monitoring the characteristics of environments has become the use of sensors connected to battery powered microcontroller based data loggers [1]–[3]. Unlike the older custom built data loggers, the attractiveness of using battery powered microcontroller based data loggers is their simplicity in design and low cost - allowing them to be easily utilised economically in a wide range of applications without being cost heavy.

When considering which microcontroller to use for such applications, researchers are spoilt for choice with many cheap and easily programmable controllers available. Recently, due to the ever growing collection of Arduino based devices [4], the Atmega328P is fast becoming a recognised microcontroller that is at the heart of many private and industrial based research projects due to ease of use and flexible design [5], [6].

The associate editor coordinating the review of this manuscript and approving it for publication was Vyasa Sai.

Despite widespread use in a range of applications such as combat submarine navigation [2], solar panel monitoring [5], wind turbine control [7], livestock behavioural tracking [8] and radiation monitoring [9], there is yet to be a universally established method for acquiring and saving data in microcontroller projects and this can be seen when examining the variety of topologies and their implementations. The three most common memory storage devices for data logging are the electronic erasable and programmable read-only memory (EEPROM), the secure digital (SD) card, or the universal serial bus USB) flash drive. All of these can be implemented into microprocessor based logging projects, but offer varying properties. EEPROMs usually offer superbly low current levels but are limited to memory capacities in the MB range. USB drives offer memory capacities up to 100's of GB, but use a considerable amount of power; typically drawing 20-40 mA when idle [10]. SD cards are the best of both worlds, with GB saving capacities, fast saving times in the mA range and are capable of begin reduced to μA power consumption when not in use.

TABLE 1. Comparison of microcontrollers with SPI interface.

Microcontroller	Clock (MHz)	I_{active} (mA)	I_{sleep} (uA)	Memory (KB)	SRAM (B)
Atmega328P [11]	8	3.8	0.096	32	2048
Atmega2560 [12]	4	3.2(3)	<1(3)	256	8192
ATtiny1614 [13]	10	3.1(3)	0.1(3)	16	2048
PIC18F45Q10 [14]	16	1.3(3)	0.6(3)	32	2048
LPC1768FBD [15]	100	45	0.04	512	32000
ATSAMD21G18 [16]	16	3.3	0.1	256	32000

List of suitable microcontrollers for SD card data logging applications and their relevant parameters. Active and sleep currents are typical values at the specified clock speed from the datasheet at room temperature at 3.3 V unless specified otherwise.

SD cards are the most convenient method for saving data in data logger projects that require monitoring periods of months or years. Despite being less power hungry than USB drives, they are typically the most power hungry aspect of a project [17] often limiting the minimum achievable current to the mA's range if not properly initialised and thus limiting the lifetime of battery powered loggers. However, there are many methods by which the power consumption of an SD card can be reduced by orders of magnitude. These include reducing the operating voltage from 5 to 3.3 V, lowering the operating frequency, only powering the SD card during a save event and so on. However, the applicability of these various techniques in microprocessor based data logger is not well studied. In previous work [18], an onboard EEPROM was used to reduce the frequency by which data was written to an SD card thereby significantly reducing the average current consumption of the circuit although no comparison was made to the current consumption without the use of the EEPROM. To the authors knowledge, this is the only attempt to actively address the power consumption of SD based data loggers and a detailed comparison of the effects of altering the supply voltage, and the frequency of the microcontroller, sensor measurements and number of measurements before saving, whether compared individually or collectively, has yet to be performed. As such, there is no well-established low power method for saving data to SD cards in microprocessor based data loggers which can cause long delays during the development of data logging projects.

In this paper, in order to maximise the lifetime of battery powered data logging projects, the power consumption of an SD card and the various methods by which to reduce the overall power consumption of the data logger circuit are compared. Prior to comparing the power consumption levels with the SD card, the power consumption of an Arduino UNO R3 and a bare Atmega328P are compared directly to determine the minimal achievable current for each topology, from which, the UNO is shown to be limited to a maximum battery lifetime of 3 days whilst the bare microcontroller can theoretically exceed 10 years. Following this, using an example project as a baseline, the power consumption of a data logging project that wakes every 2 seconds to measure a single sensor is compared at 5 V at 16 MHz and 3.3 V at 8 MHz. By analysing the transient waveforms of each topology, a significant increase in the average current consumption can be seen following the introduction of an SD card. Finally, it is shown how the power consumption of the data logging project can be reduced significantly using a power control MOSFET and a threshold string-length method, reducing the original power consumption by over an order of magnitude, and thus greatly increasing the maximum achievable lifetime of the data logging circuit.

A. CHOICE OF MICROCONTROLLER

A wide family of micrcontrollers exists that can be used for monitoring environmental, industrial, or commercial data logging applications. The two most common microcontrollers that are used for data logging applications are the Atmega328P [1], [5] and the Atmega2450 [2], [3]. In order to read/write data from/to an SD card, a microcontroller must have an SPI interface and a sufficient flash and SRAM memory size. A comparison of commonly used microcontrollers that are suitable for data logging applications can be seen in Table 1.

Of all the microcontrollers, considering the active state current, it can readily be seen that the active current draw from the LPC1768FBD is over an order of magnitude greater than all over microcontrollers but achieves the best sleep current of 40 nA. Second to the LPC1768FBD is the Atmega328P, although this also has the second greatest current consumption in the active on-state. In terms of memory size, all of the microcontrollers listed are capable of supporting SPI interface with an SD card although it can be seen that the LPC1768FBD and the ATSAMD21G18 provided the greatest flash and SRAM sizes which could be needed for larger datalogging applications.

As it will be shown throughout this work, the overall limiting factor to battery life in data loggers comes from the introduction of the SD card which raises the active and sleep current of the circuit to peak values of 80 mA and 0.8 mA respectively which, excluding the active current of the LPC1768FBD, is far greater than the active and sleep current of each microcontroller. Here, we focus on programming the Atmega328P due to it's commercial familiarity and as this allows for a direct comparison of the current draw from the Arduino UNO board and the microcontroller. Despite this, each microcontroller listed in Table 1 has been shown to be suitable for datalogging applications [9], [19]–[21] and the methods presented here can be applied to all microcontrollers to maximise the battery life of data logging applications with SD cards.

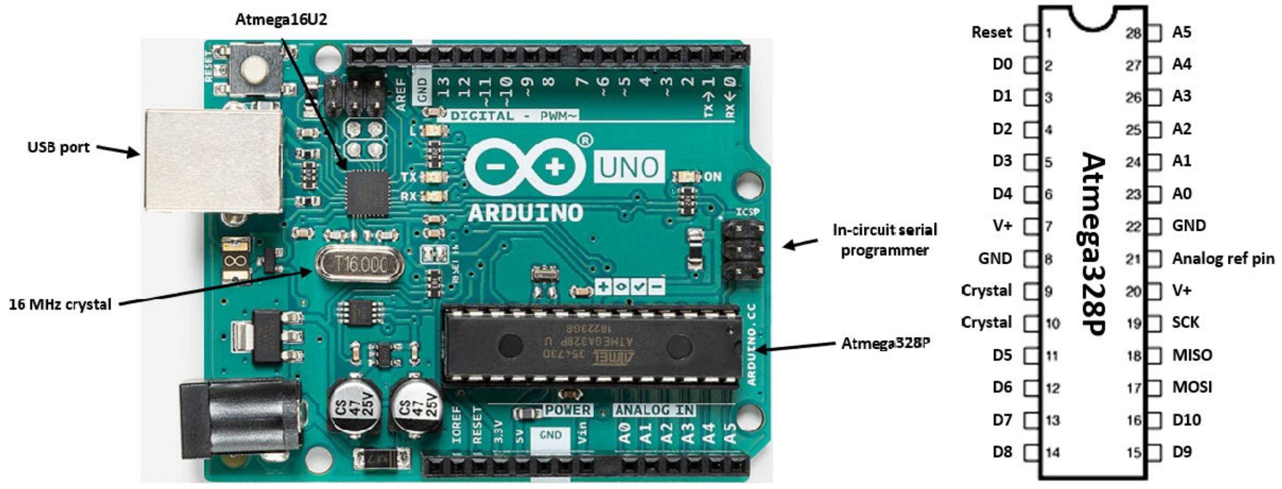

FIGURE 1. **Arduino UNO R3 and Atmega328P pin out. Digital pins 11-13 are used for the serial peripheral interface (SPI) connections to the micro SD card.**

II. ARDUINO UNO AND THE Atmega328P

The Arduino boards were first introduced in 2005 resulting from the Interaction Design Institute Ivrea graduate programme in Ivrea, Italy. The standard boards essentially act as a wrapper for Atmel microprocessors allowing hobbyists and researchers to quickly develop microcontroller projects. A picture of the current iteration of the Arduino UNO board [22] can be seen in Fig. 1 which is based on the Atmega328P microcontroller.

The standard Arduino Uno board comes attached with multiple discrete devices including capacitors, resistors and voltage regulators as well as other components which are crucial for programming and operation such as the Atmega16U2 IC/USB controller, voltage regulators and 16 MHz crystal. When developing projects, the majority of these components are required for uploading sketches to the Atmega328P microcontroller, but once programmed, the majority of these components are not required.

The attractiveness of the UNO in comparison to bare Atmega328P chip is in the ease of programming. Due to the Atmega16U2 IC/USB microcontroller, sketches can quickly be uploaded from the computer to the Arduino UNO board in a matter of seconds. In contrast, when using the bare Atmega328P microcontroller, sketches must be uploaded using either a separate Arduino as an in-circuit serial (ISP) programmer, or through a FTDI programmer [23] a technique which can be prone to time-out failures and potential chip damage if not wired up correctly.

As such, the UNO is definitely the better choice for introductory users to projects, but the suitability of using the UNO for battery powered projects, especially when compared to using the Atmega328P as a standalone chip, is yet to be compared.

III. METHOD

In order to determine the power consumption of the Arduino UNO board in comparison to the standalone Atmega328P chip, the effects of the various power saving methods covered in the Atmega328P datasheet were compared [11]. Firstly, to ascertain the total power consumption of the discrete components on the UNO board, the Atmega328P was powered at 5 V with a 16 MHz crystal oscillator. Following this, the power was reduced to 3.3 V (the typical voltage for low power SD applications) and the crystal was replaced with an 8 MHz crystal. For all cases, the power was supplied from a Keithley 2410 sourcemeter and the current was measured from the on screen value once a steady state value had been achieved.

Following the power consumption comparison, an Adafruit MicroSD breakout board+ was used as the interface between the SPI pins of the Arduino/Atmega328P chip and the micro SD card [24]. The advantage of using the Adafruit breakout board is that it provides both 5 V and 3.3 V power lines depending on the logic level of the circuit which might be used for an associated sensor.

In order to determine the power consumption of the SD card during writing events, an Adafruit PCF8523 real time clock (RTC) was used to trigger the Arduino every 2 seconds. The RTC was powered from a separate coin cell to ensure that the RTC does not contribute to the overall power consumption of the circuit. For each topology, the power consumption and time taken during SD write events and sleep mode is measured using an RIGOL DS1074 oscilloscope.

With all of the waveform diagrams, the total power consumption is compared with a focus towards determining the lifetime of a battery powered data logger.

IV. POWER CONSUMPTION: ARDUINO VS Atmega328P

In order for the added power consumption of an SD card to be contextualised, the power consumption of the Arduino board and Atmega328P microcontroller must be determined. For a fair comparison, the Atmega328P chip was measured twice, once at an operating voltage of 5 V with a crystal frequency of 16 MHz, mimicking the operating conditions

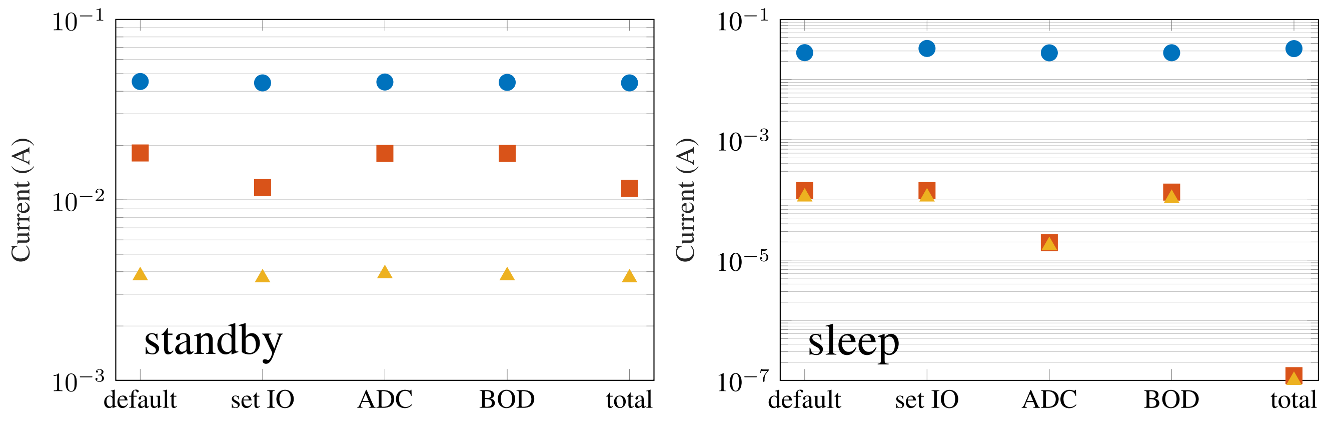

FIGURE 2. Current drawn from power supply from the Arduino UNO (●), Atmega328P at 5 V at 16 MHz (■), and the Atmega328P at 3.3V at 8 MHz (▲) during standby (left) and "power-down" sleep mode (right).

of the Arduino UNO, and once again at 3.3 V at 8 MHz for comparison.

For each circuit topology, the draw current of the circuit was determined using a script that ran the following sequence:

1) setup input/output (IO) pins
2) turn on LED
3) wait 1 second
4) turn off LED
5) wait 1 second
6) go to sleep

where the current was measured at steps 5) and 6).

By reading through the Atmega328P datasheet [11], a considerable number of recommendations on how to use the microcontroller can be found for power saving methods. In order to implement to majority of the power reduction techniques, the microcontroller must be put into the "power-down" sleep mode by setting the correct bits in the sleep mode control register (SMCR):

```
SMCR |= bit(SM1) | bit(SE);
__asm__ __volatile__("sleep");
```

from which the device can only be re-activated if an interrupt has been set up on digital pins 2 or 3 or if the reset pin is activated. In power-down mode, all internal oscillators are halted preventing operation of all the internal registers effectively halting all operations.

In further reducing power consumption, the Atmega328P datasheet recommends further options to help reduce power consumption during sleep mode including:

1) Disable all digital inputs on the analogue input pins and set all unused digital pins to outputs
2) Disable the ADC through the ADCSRA register
3) Disable brown out detection (BOD) using the MCUCR register (note: this step must be performed just before executing the "sleep" command)

all of which can be implemented using:

```
// Set digital pins 0-13~to output
  for(int i = 0;i<14;i++){
    pinMode(i,OUTPUT);
  }
// Disable digital pins 14-19~on ADC
  DIDR1 = bit(AIN1D)|bit(AIN0D);
  DIDR0 = bit(ADC5D)|bit(ADC4D)|bit(ADC3D)|
    bit(ADC2D)|bit(ADC1D)|bit(ADC0D);

// Turn off ADC
  ADCSRA &=~bit(ADEN);

// Disable BOD
  MCUCR |= bit(BODS) | bit(BODSE);
  MCUCR = (MCUCR &~bit(BODS)) | bit(BODSE);
```

When the microcontroller has been triggered, it is important to re-enable the ADC in order to measure any sensors using:

```
  ADCSRA  |= bit(ADEN);
```

If this is not performed, any subsequent ADC measurements will return the last value that was measured by the ADC before being disabled.

A comparison of the current consumption of the three microcontroller topologies can be seen in Fig. 2 where each power saving technique is also compared. It can be seen that, when the device is in the active mode and is waiting for commands, there is no difference in the current consumption from turning off the ADC and disabling the BOD, but setting all of the IO pins reduces to current consumption of the Atmega328P by close to 50 %.

In sleep mode, the reverse is true, and it can be seen that setting the IO pins has no effect on the current consumption in sleep mode, but disabling the BOD and the ADC reduces the current draw by 2 and 3 orders of magnitude for the Atmega328P. For the Arduino UNO, the parasitic leakage from the discrete components can be seen clearly. Even when all of the various sleep modes are combined, the minimal achievable output current is limitted to 28 mA. In contrast, when combining all of the power saving techniques, the minimal achievable sleep current for the Atmega328P microcontroller was measured at 120 and 97 nA respectively, which is ideal for battery applications.

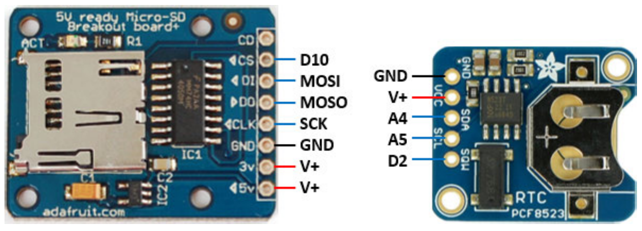

FIGURE 3. Connections from adafruit SD and RTC modules. On the Arduino/Atmega328P, the analogue pins A4 and A5 also act as the serial data (SDA) and serial clock (SCL) pins respectively.

V. SAVING TO THE SD CARD

When using the SD card, the standard method for saving data to the card is to initialise the SD card during the setup phase of the programme, from which, any data can then be saved to the SD card in a user specified file format.

To compare the power consumption of the SD card, the SD card was measured on the Arduino UNO and both Atmega328P circuits. For the test, the following pseudocode was executed:

```
1) Setup IO, RTC and initialise SD card
2) Begin main loop:
      a) Read voltage at analogPin A0
      b) Save data to SD card
      c) go to sleep
```

from which, the main loop was repeated whenever the microcontroller was triggered by the RTC.

The setup for the circuit can be seen in Fig 3. During the setup phase, the PCF8523 is programmed to send a falling edge pulse to digital pin 2 of the Arduino/Atmega328P. When the microcontroller detects the pulse, it wakes up and repeats steps 2) - 4). A plot of the recorded waveforms from the Arduino and Atmega328P circuits can be seen in Fig 4. The measured waveforms include the falling edge trigger on digital pin 2 from the RTC, the voltage on the chip select (CS) pin of the SD card, and the calculated current draw from each circuit. In order to measure the current draw of the circuits for each topology, a 10 Ω shunt resistor was placed in series between the power supply and the circuit. The voltage drop over the shunt resistor was measured, from which, the resulting current draw of the circuit was calculated using

$$I = \frac{V_{\text{supply}} - V_{\text{measured}}}{10} \tag{1}$$

where V_{supply} was increased from 5 and 3.3 V to 5.5 and 3.5 V to minimise the effect of the shunt resistor voltage drop on the circuit operation.

From the data, it can be seen that there is a clear delay incurred when operating the microcontroller at 8 MHz as opposed to 16 MHz. The 16 MHz devices take 9-10 ms to complete writing to the SD card whilst the 8 MHz device takes 15 ms. Aside from the writing to the SD card, it can be seen that there is an increased delay between the falling edge of the RTC trigger and the falling edge of the CS. This is expected, as the microcontroller is running at half the clock speed.

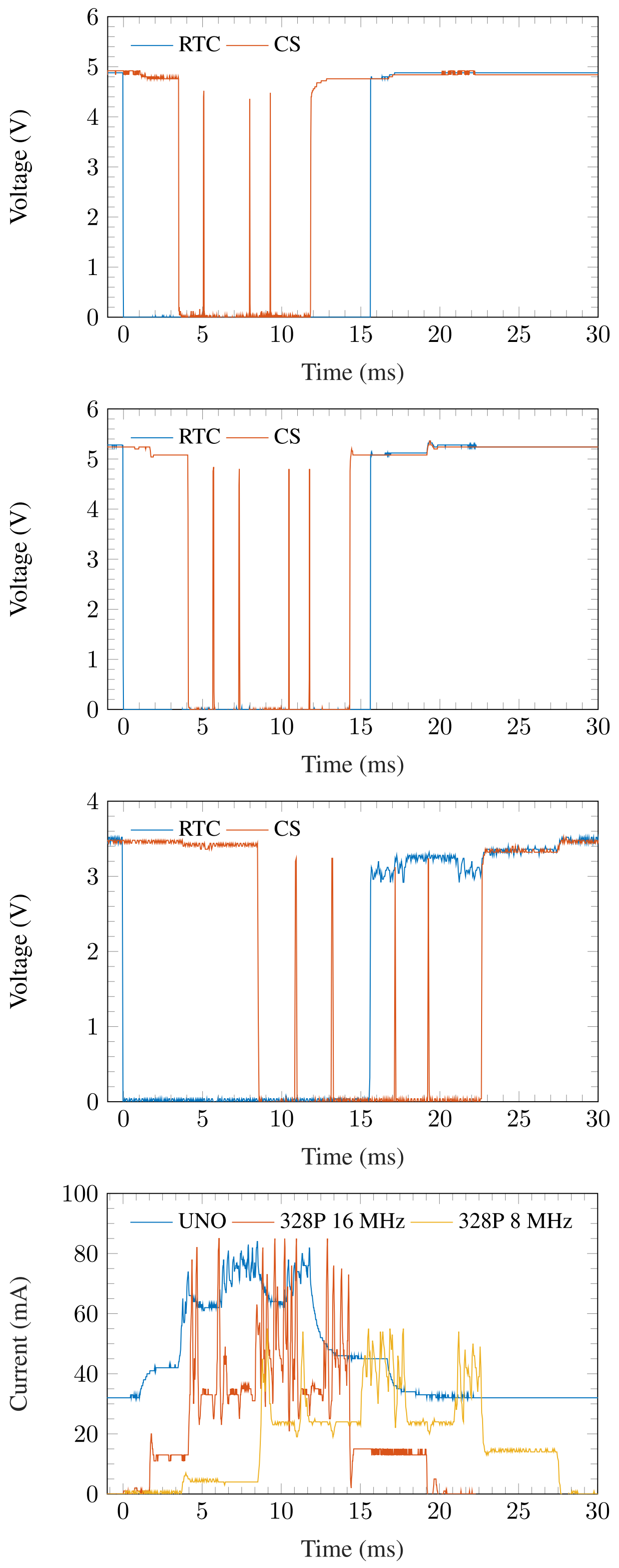

FIGURE 4. From top to bottom, transient voltage response of the UNO, Atmega328P at 5 V at 16 MHz, the Atmega328P at 3.3 V at 8 MHz and the calculated transient current response of the three circuits.

Observing the transient current waveforms, the differences in current consumption between the three circuits can be seen clearly. During the writing phase, both the UNO and

TABLE 2. Topology lifetime calculation.

Microcontroller	Current		Lifetime	
	No SD	SD	No SD	SD
Arduino UNO	28 mA	30 mA	60 H	56 H
328P 5 V 16 MHz	120 nA	0.8 mA	>10 Y	87.5 D
328P 3.3 V 8 MHz	96 nA	0.75 mA	>10 Y	97.3 D

Lifetime calculation for each topology when powered by a battery with a capacity of 2400 mAH. Y = years, D = days, H = hours.

the Atmega328P draw close to 80 mA at peak whilst the Atmega328P running at 3.3 V at 8 MHz draws 55 mA at peak. Aside from the current consumption during the SD write phase, the advantage of moving from the UNO to the Atmega328P is also indicated in the data. Although the UNO has one of the shortest writing times, the current level rests at 33 mA even when the device has entered sleep mode. On the other hand, even with the introduction of the SD card reader, the Atmega328P at 5 V at 16 MHz and 3.3 V at 8 MHz have sleep currents of 0.8 and 0.75 mA respectively.

VI. CIRCUIT LIFETIME

Although sleep currents of 0.8 and 0.75 mA for the microcontroller topoligies are two orders of magnitude lower than the sleep current of the Arduino UNO, this is still much greater compared to the sleep current without the SD card reader. When considering data logging applications, the lifetime of a battery can be calculated using:

$$L = 0.7\frac{\text{Battery capacity}}{I_{\text{load}}} \tag{2}$$

where L is the battery lifetime and I_{load} is the current draw from the battery by the load [25]. Here, we also use the commonly used pre-factor of 0.7 [26] for safety in the battery lifetime calculations to take into account the potential increase in circuit current consumption due to temperature, noise, and so on. When considering data logging applications, the current draw from the load becomes time dependent and must be calculated using

$$I_{\text{load}} = \frac{1}{T}\int_0^T i(t)\,dt \tag{3}$$

where $i(t)$ is the time dependent current draw of the circuit and T is the period. When considering the total current from the supply, as the current draw from circuit during sleep is time independent, the current can be split into two separate terms such that

$$\begin{aligned} I_{\text{load}} &= \frac{1}{T}\left(\int_0^{t1} I_{\text{sleep}}\,dt + \int_{t1}^{T} I_{\text{active}}\,dt\right) \\ &= \frac{1}{T}\left(I_{\text{sleep}}t1 + \int_{t1}^{T} I_{\text{active}}\,dt\right) \end{aligned} \tag{4}$$

where $t1$ is the time at which the microcontroller wakes to measure the device, I_{active} and I_{sleep} are the current consumptions of the circuit when the microcontroller is awake and in the power-down sleep mode respectively. It can readily be seen from Eq. (4) that as the measuring period of the circuit increases, so too will the lifetime of the circuit.

For the three circuits considered here, the introduction of the Micro SD card reader raised the minimum sleep current for the Arduino UNO, Atmega328P at 5 V at 16 MHz and the Atmega328P at 3.3 V at 8 MHz from 28 mA, 120 nA and 97 nA to 30 mA, 800 μA and 750 μA respectively. A comparison of how this effects the claculated lifetimes can be seen in Table 2. Although the increase in sleep current is negligible for the Arduino UNO, both the microcontroller topoligies experience an increase in leakage current by 3 orders of magnitude. Although the lifetime for the UNO will have a negligible reduction, using Eq. (2), the lifetime of the two Atmega328P circuits experience a reduction in lifetime from over 10 years each to 97 and 87 days when using a 2400 mAH battery.

It is important to note that the lifetime calculated for the microcontrollers without the SD card are based on the sleep current of the microcontroller circuits with no external measurement/saving components, as such, these values should be contextualised as a maximum achievable lifetime for the microcontrollers and is not realistically achievable for most applications. Despite this, the reduction in the lifetime of the microcontroller circuits from over 10 years to close to 3 months is significant and detrimental for long term applications.

For a good selection of applications, a lifetime of 3 months when powered from 2 batteries is acceptable as this can be doubled to half a year by placing another set of batteries in parallel and so on, but for long term data gathering experiments such as sub-sea data loggers, this is less than ideal from a physical device size and cost point of view.

VII. MOSFET POWER SAVING

The added leakage current from the SD card reader reduces the potential lifetime of the Atmega328P microcontroller circuits from over 10 years to 3 months. The cause of the added leakage is due to the SD card remaining powered at all times even when the microcontroller is in sleep mode. To remove this leakage from the sleep mode current, a BS170 N-channel MOSFET [27] was placed between the ground pin of the SD card reader and ground of the circuit.

In order to allow the SD card to be powered using a MOSFET, the card must be initialised on every wake cycle [18]. As such, the psuedocode for the microcontrollers was changed to:

```
1) Setup IO and RTC
2) Begin main loop:
    a) Read voltage at analogue Pin A0
    b) Initialise SD card
    c) Save data to SD card
    d) go to sleep
```

from which, the main loop was repeated whenever the microcontroller was triggered by the RTC. The effect of using the MOSFET on the UNO was not explored as the minimum current achievable with the board cannot be reduced below 28 mA.

A plot of the waveforms for an SD save event after the SD card has been powered using the MOSFET can be seen in Fig. 5. From the data, it can be seen that the period of time that the microcontroller is awake has increased from ~20 ms to over 150 ms for both the 16 MHz and 8 MHz microcontrollers. The cause in the extended wake period is due to the SD card initialisation phase that must be performed at the beginning of every wake cycle.

With the MOSFET, the sleep current of the microcontrollers drops from 800 and 750 μA to 21.1 and 18.6 μA respectively. Although the sleep current is much lower, the average current drawn from both circuits exceed the previous setup due to the extended SD active time. From Eq. 4, the predicted lifetime of this setup is calculated at 16 and 24 days for the Atmega328P at 5 V at 16MHz and 3.3 V at 8 MHz respectively, a reduction by over two months compared to leaving the SD card powered at all times.

A. MEASURING PERIOD AND PERIODIC SAVING

As shown in the previous section, when using a measuring period of 2 seconds, the average current draw from the microcontrollers increases when using a MOSFET to power the SD card reader due to the extended period of time needed to initialise the SD card. In order to deal with this, two techniques can be employed; increase the saving period and/or save data to the SD card periodically.

The average current draw from the circuit is given by Eq. 4 and it can be seen that the battery life can be extended from reducing the measuring frequency, although this can reduce the quality of data from a research standpoint. Aside from reducing the measuring frequency, another technique that can be used is to take advantage of the Atmega328P's ability to store variables between sleep modes in the internal static random access memory (SRAM).

When the Atmega328P enters sleep mode, it stores all of the current variables in memory and all of these variables are available for use during the next wake cycle. As such, it is possible to save multiple measurements in a single event as apposed to writing to the card after awake RTC wake cycle.

To demonstrate this, the code on the microcontrollers was adapted to only save to the SD card after a global string variable had exceeded a certain length, as such, the psuedocode was changed to:

```
1) Setup IO and RTC
2) Declare global string: dataString
3) Begin main loop:
    a) Measure analogue pin A0
    b) Append time and measurement to dataString
    c) if length(dataString) > safety cap: initialise and
       save to SD card and clear the variable for next
       RTC cycle
    d) Go to sleep
```

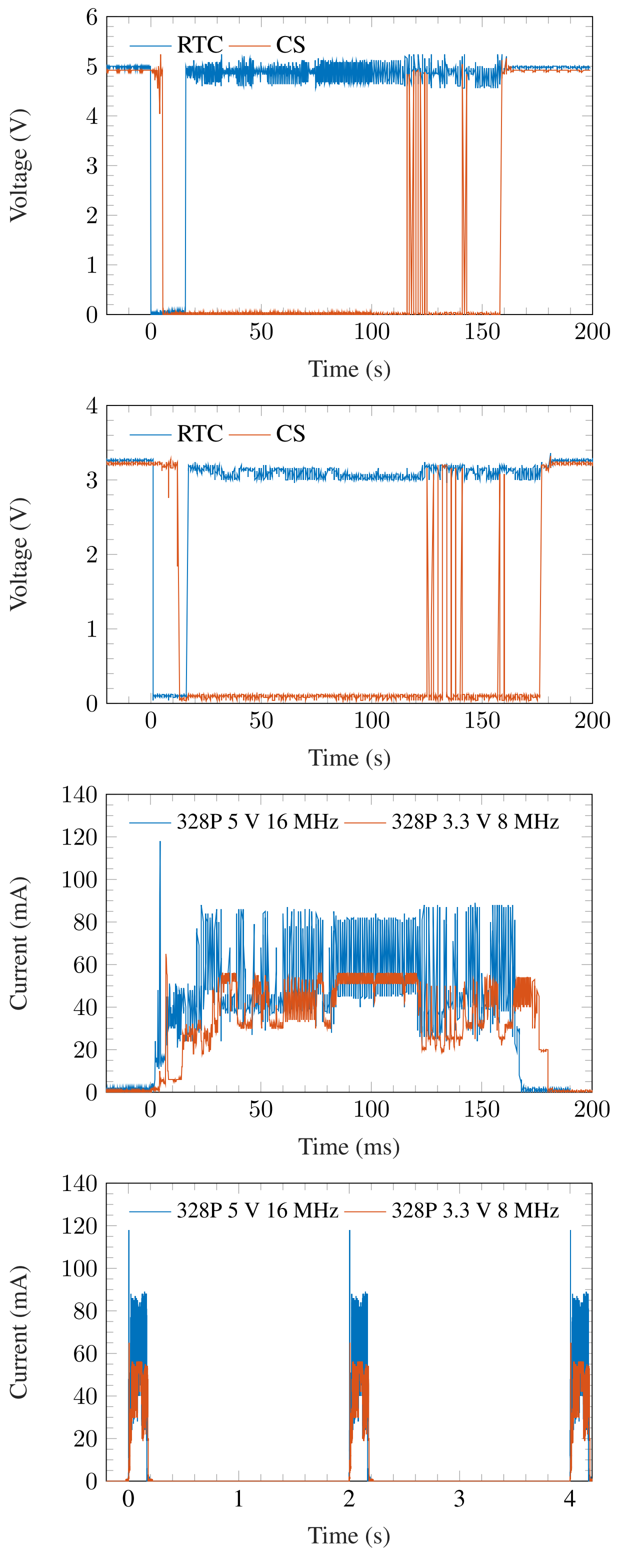

FIGURE 5. From top to bottom, transient voltage response of the Atmega328P at 5 V at 16 MHz, the Atmega328P at 3.3 V at 8 MHz and the calculated transient current response of the two circuits.

now, the data will only be saved to the SD if the length of **dataString** exceeds a threshold value which can be chosen by the user. As an example, we have set the safety cap to ensure

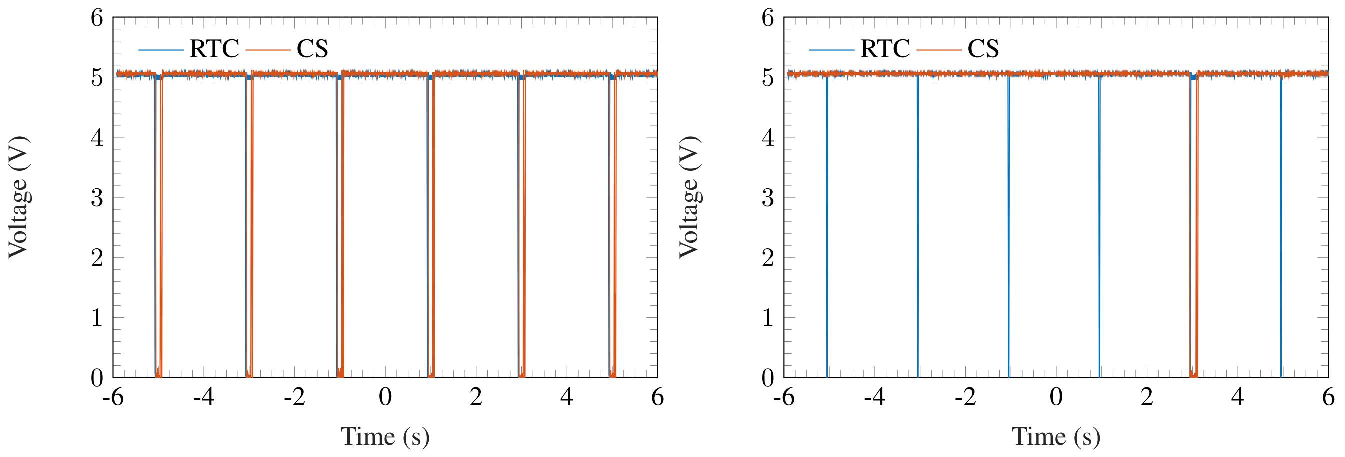

FIGURE 6. Comparison of the CS patterns on the SD card when using the MOSFET to save every time (left) compared to saving once every 6 RTC cycles (right).

that the Atmega328P will only save once every 6 RTC cycles, as illustrated in Fig. 6. From both plots, it can be seen that by using the threshold saving technique, the CS of the SD only becomes active once every 6 RTC cycles, greatly reducing the average load current of the circuit.

By using the threshold string length method, the total trigger time of the device is effectively increased by a factor equal to the number of clock cycles needed to save to the SD. Using the MOSFET, the average current consumption for the Atmega328P at 5 V at 16 MHz and at 3.3 V at 8 MHz decreased from 4.31 and 3.43 mA to 750 and 590 μA respectively, which is a lower average current compared to keeping the SD powered at all times.

B. LIFETIME ESTIMATION

Throughout, the assumption of the setup here is that the microcontroller would be used for applications that require measuring sensors every 2 seconds. In reality, data logging applications have been able to gather valuable data from environments using much longer periods of measuring every 30 seconds or even 1-2 minutes [28], [29].

The effect of increasing the measuring time period between each measurement on the battery lifetime can be determined using Eq. (2).

For the microcontrollers, the mean mean current and time required to save data to the SD card when using the MOSFET was 8.74 and 6.82 mA for 0.23 and 0.24 s for the Atmega328P at 5 V at 16 MHz and at 3.3 V at 8 MHz respectively. Using these values, the mean load current of the two circuits can be calculated using:

$$I_{\text{load}}(5\ V) = \frac{1}{NT}\left[(NT - 0.23)2.11 \times 10^{-6} + 0.00201\right] \tag{5a}$$

$$I_{\text{load}}(3.3\ V) = \frac{1}{NT}\left[(NT - 0.24)1.86 \times 10^{-6} + 0.00164\right] \tag{5b}$$

where T is the period and N is the number of RTC cycles per saving to the SD. Using Eq. (5), the lifetime of the device can also be estimated using Eq. (2). A plot of the mean load current and resulting calculated lifetimes can be seen in Fig. 7. From the data, it can be seen that using the SRAM within the Atmega328P to store data over multiple RTC cycles increases the lifetime of the device by an order of magnitude when the measuring period is below 10 seconds. It can also be seen that, as the RTC trigger period is increased, the reduction in load current and resultant increase in battery lifetime begins to plateau above 15 seconds when saving every time or after 6 RTC cycles. Comparing the devices directly, it can be seen that moving from 5 V at 16 MHz to 3.3 V at 8 MHz does not result in a significant increase in lifetime. From 1 to 60 seconds, reducing the operating voltage and frequency from 5 V at 16 MHz to the 3.3 V at 8 MHz increases the lifetime of the circuit by 15 to 20% in comparison to using the 5 V supply. Although this increase is not significant, there is still a significant advantage to using the Atmega328P at 8 MHz near 3.3 V in that it can be powered using 2 AA batteries as apposed to using 4 AA batteries with a 5 V regulator.

For data logging applications, it can be seen that the battery lifetime is increased significantly when using the SRAM within the Atmega328P to ensure that the data is periodically written to the SD card. Analysing the lifetime data in Fig. 7, in order to ensure measurements for one year, the minimum RTC period when saving every time ($N = 1$) is 12 seconds for the microcontroller at 5 V at 16 MHz and 10 seconds at 3.3 V at 8 MHz. When using the SRAM to reduce the saving period to every 6 RTC cycles, the minimum measurement period can be reduced to 2 and 1.7 seconds for the two circuits whilst still ensuring a battery lifetime of one year.

C. SENSOR MEASURING TIME CONSIDERATIONS

Within this work, current draw of the circuit when writing data to the SD card has been assumed to be the only significant source of current when the microcontroller is awake. Depending on the application, some external sensors require

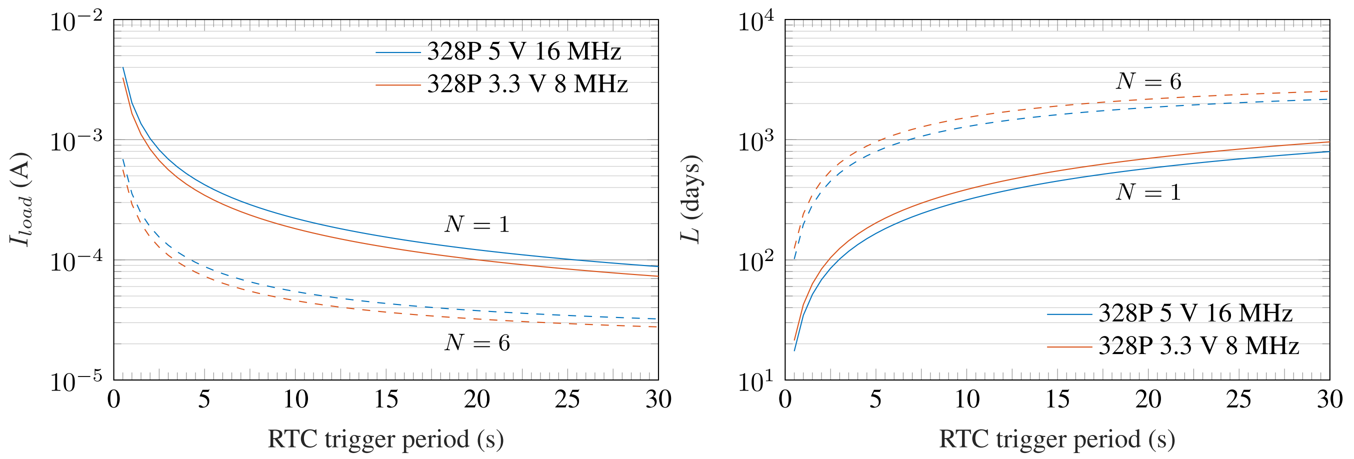

FIGURE 7. Calculated mean load current and resultant lifetime of 2400 mAH battery for the Atmega328P microcontroller circuits when using the MOSFET to power the SD card during write events. The factor N indicates the number of RTC cycles per SD write event.

stabilisation periods of 1-2 seconds and this will significantly effect the lifetime calculations here.

Aside from external sensor stabilisation, the added time for additional ADC measurements can also increase the total active state time. For the Atmega328P, a single ADC measurement will take 25 ADC clock cycles after the ADC is switched on and 13 cycles for every subsequent measurement. By default, the ADC of the Atmega328P is set to operate at 1/128th of the crystal oscillator frequency, that is, the operating frequency of the ADC is 125 and 62.5 KHz when using the 16 and 8 MHz crystal oscillator. With this, the total time for the ADC to perform a measurement at any pin would be:

$$t_{16\text{ MHz}} = \frac{25 + 13(A-1)}{125\text{ KHz}} \tag{6a}$$

$$t_{8\text{ MHz}} = \frac{25 + 13(A-1)}{62.5\text{ KHz}} \tag{6b}$$

where A is the number of measurements assuming it has just been enabled. As such, a single measurement for every wake cycle will take 200 and 400 μs when using the 16 and 8 MHz crystal oscillators and will increase by 104 and 208 μs for each subsequent measurement. Although this time is negligible in comparison to the length of time to write to the SD card, this can quickly add up if more sensor are added and if multiple ADC measurements are needed on sensors that require averaging to reduce noise.

The time required to measure sensors and whether or not any averaging techniques are requires is beyond the scope of this work as this is both application and sensor dependent. As well as this, the added current draw from using sophisticated sensors with built in operational amplifiers and filters could contribute to the overall current consumption of the circuit and total wake time. Once again, this is beyond the scope of the work covered here, but these aspects should be taken into consideration when determining the lifetime of data logging applications and can easily be included into the calculations presented here if the current and active time of the sensors are known.

VIII. CONCLUSION

In this paper, the lifetime of an Arduino UNO, Atmega328P at 5 V at 16 MHz and the Atmega328P at 3.3 V at 8 MHz with and without an SD card reader has been measured. Without the SD card, the Atmega328P was capable of reaching sleep currents of 120 and 96 μA when powered at 5 V at 16 MHz and 3.3 V at 8 MHz respectively, whilst the UNO was only capable of achieving a minimum current of 28 mA when in sleep mode.

Following the introduction of an SD card reader, it was found that the sleep current for the Arduino UNO, Atmega328P at 5 V at 16 MHz and the Atmega328P at 3.3 V at 8 MHz increased to 30, 0.8 and 0.75 mA respectively, dramatically reducing the maximum achievable battery lifetime. By introducing a BS170 power control N-channel MOSFET, the sleep current for the Atmega328P microcontroller at 5 V at 16 MHz and 3.3 V at 8 MHz was reduced to 21.1 and 18.6 μA, although the increased time required to initialise the SD to allow the saving of data resulted in an increase in mean current draw of the circuit to 4.31 and 3.43 mA respectively.

With further analysis, it was found that the increase in load current was due to the increased SD write time which can be circumvented by increasing the period between measurement and/or taking advantage of the Atmega328Ps built in SRAM. By using both of these methods it was found that the Atmega328P at 5 V at 16 MHz and 3.3 V at 8 MHz is capable of logging data for a period of 1 year if the measuring period is reduced to 12 and 10 seconds respectively. By taking advantage of the SRAM within the microcontroller, this can be further reduced to 2 and 1.7 seconds respectively. In future work, a comparison between the Atmega328P and alternative microcontrollers with superior SRAM and faster clock speeds should be performed as this could achieve superior characteristics to that of the Atmega328P.

With the techniques presented here, a standard method for producing a microcontroller based data logger that can record data for 1 year down to a measurement period of 1.7 seconds has been achieved. The methods here are easily repeatable and only require a microcontroller with SPI interface, 2/4 AA batteries, 8/16MHz crystal, RTC and the Adafruit SD card reader with SD card to be recreated - all of which can be purchased for ∼£30.

REFERENCES

[1] J. P. Grinias, J. T. Whitfield, E. D. Guetschow, and R. T. Kennedy, ''An inexpensive, open-source USB arduino data acquisition device for chemical instrumentation,'' *J. Chem. Edu.*, vol. 93, no. 7, pp. 1316–1319, Jul. 2016, doi: 10.1021/acs.jchemed.6b00262.

[2] V. Rahmawati, A. Affandi, I. Arifin, and M. Asrofi, ''Acquisition and data logging inertial navigation system (INS) on combat submarine using arduino-SD card,'' *IPTEK J. Proc. Ser.*, vol. 0, no. 3, p. 58, Aug. 2019, doi: 10.12962/j23546026.y2019i3.5843.

[3] G. Vitale, S. Scudero, A. D'Alessandro, A. Pisciotta, R. Martorana, and P. Capizzi, ''New ultraportable data logger to perform magnetic surveys,'' in *Proc. Int. Symp. Adv. Electr. Commun. Technol. (ISAECT)*, 2019, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/document/9069730, doi: 10.1109/ISAECT47714.2019.9069730.

[4] K. A. Rodriguez-Vasquez, A. M. Cole, D. Yordanova, R. Smith, and N. M. Kidwell, ''AIRduino: On-demand atmospheric secondary organic aerosol measurements with a mobile arduino multisensor,'' *J. Chem. Edu.*, vol. 97, no. 3, pp. 838–844, Mar. 2020, doi: 10.1021/acs.jchemed.9b00744.

[5] M. S. Hadi, A. N. Afandi, A. P. Wibawa, A. S. Ahmar, and K. H. Saputra, ''Stand-alone data logger for solar panel energy system with RTC and SD card,'' *J. Phys., Conf. Ser.*, vol. 1028, Jun. 2018, Art. no. 012065, doi: 10.1088/1742-6596/1028/1/012065.

[6] N. A. P. Floretes, ''A decentralized data logger system for water related disaster with wireless mesh topology,'' *Int. J. Integr. Eng.*, vol. 12, no. 1, pp. 108–114, Jan. 2020. [Online]. Available: https://publisher.uthm.edu.my/ojs/index.php/ijie/article/view/4044

[7] S. Murti, P. Megantoro, G. D. B. Silva, and A. Maseleno, ''The design and analysis of DC electrical voltage-current datalogger device implemented on wind turbine control system,'' *J. Robot. Control*, vol. 1, no. 3, pp. 75–80, 2020, doi: 10.18196/jrc.1317.

[8] L. Nóbrega, P. Gonçalves, M. Antunes, and D. Corujo, ''Assessing sheep behavior through low-power microcontrollers in smart agriculture scenarios,'' *Comput. Electron. Agricult.*, vol. 173, Jun. 2020, Art. no. 105444, doi: 10.1016/j.compag.2020.105444.

[9] L. Chen, Y. Jing, Q. Zheng, and F. Xiao, ''Fault diagnosis device for nuclear radiation monitoring system,'' in *Proc. IEEE 4th Adv. Inf. Technol., Electron. Autom. Control Conf. (IAEAC)*, vol. 1, 2019, pp. 2012–2018. [Online]. Available: https://ieeexplore.ieee.org/document/8998041, doi: 10.1109/IAEAC47372.2019.8998041.

[10] K. O'Brien, D. C. Salyers, A. D. Striegel, and C. Poellabauer, ''Power and performance characteristics of USB flash drives,'' in *Proc. Int. Symp. World Wireless, Mobile Multimedia Netw.*, 2008, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/document/4594868, doi: 10.1109/WOWMOM.2008.4594868.

[11] *Datasheet for Atmega328P and Variants*. Accessed: Jul. 20, 2020. [Online]. Available: www.microchip.com/wwwproducts/en/ATmega328p

[12] *Datasheet for Atmega2560 and Variants*. Accessed: Jul. 20, 2020. [Online]. Available: www.microchip.com/wwwproducts/en/atmega2560

[13] *Datasheet for ATtiny1614 and Variants*. Accessed: Jul. 20, 2020. [Online]. Available: www.microchip.com/wwwproducts/en/ATTINY1614

[14] *Datasheet for PIC18F45Q10 and Variants*. Accessed: Jul. 20, 2020. [Online]. Available: https://www.microchip.com/wwwproducts/en/PIC18F45Q10

[15] *Datasheet for LPC1768FBD and Variants*. Accessed: Jul. 10, 2020. www.nxp.com/products/processors-and-microcontrollers/arm-microcontroll%ers/general-purpose-mcus/lpc1700-cortex-m3/512kb-flash-64kb-sram-ethernet-usb-%lqfp100-package:LPC1768FBD100

[16] *Datasheet for ATSAMD21G18 and Variants*. Accessed: Jul. 20, 2020. [Online]. Available: www.microchip.com/wwwproducts/en/ATsamd21g18

[17] A. Caviezel, M. Schaffner, L. Cavigelli, P. Niklaus, Y. Buhler, P. Bartelt, M. Magno, and L. Benini, ''Design and evaluation of a low-power sensor device for induced rockfall experiments,'' *IEEE Trans. Instrum. Meas.*, vol. 67, no. 4, pp. 767–779, Apr. 2018, doi: 10.1109/TIM.2017.2770799.

[18] P. A. Beddows and E. K. Mallon, ''Cave pearl data logger: A flexible arduino-based logging platform for long-term monitoring in harsh environments,'' *Sensors*, vol. 18, no. 2, p. 530, Feb. 2018, doi: 10.3390/s18020530.

[19] M. Lambrichts, J. M. Tijerina, and R. Ramakers, ''SoftMod: A soft modular plug-and-play kit for prototyping electronic systems,'' in *Proc. 14th Int. Conf. Tangible, Embedded, Embodied Interact.*, Feb. 2020, p. 287, doi: 10.1145/3374920.3374950.

[20] D. Salave, P. Khade, J. Ghumare, R. Khalane, and R. Nikam, ''Microcontroller based detection and protection of induction motor,'' *Int. Res. J. Eng. Technol.*, vol. 5, no. 2, p. 15, 2018. [Online]. Available: https://www.irjet.net/volume5-issue2irjet.net/volume5-issue2

[21] D. Qu, B. Yang, and N. Gu, ''Indoor multiple human targets localization and tracking using thermopile sensor,'' *Infr. Phys. Technol.*, vol. 97, pp. 349–359, Mar. 2019, doi: 10.1016/j.infrared.2019.01.011.

[22] *Official Arduino UNO R3*. Accessed: Jul. 21, 2020. [Online]. Available: https://store.arduino.cc/arduino-uno-rev3

[23] *Arduino ISP Programming Tutorial*. Accessed: Jul. 21, 2020. [Online]. Available: www.arduino.cc/en/Tutorial/ArduinoISP

[24] *Adafruit MicroSD Breakout Board+*. Accessed: Jul. 21, 2020. [Online]. Available: www.adafruit.com/product/254

[25] F. Kerasiotis, A. Prayati, C. Antonopoulos, C. Koulamas, and G. Papadopoulos, ''Battery lifetime prediction model for a WSN platform,'' in *Proc. 4th Int. Conf. Sensor Technol. Appl.*, Jul. 2010, pp. 525–530, doi: 10.1109/SENSORCOMM.2010.85.

[26] *Battery Lifetime Calculator*. Accessed: Jul. 21, 2020. [Online]. Available: https://uk.farnell.com/battery-life-calculator#uk.farnell.com/batter%y-life-calculator

[27] *Datasheet for BS170 N-Channel MOSFET*. Accessed: Jul. 21, 2020. [Online]. Available: www.onsemi.com/pub/Collateral/BS170-D.PDF

[28] M. Fuentes, M. Vivar, J. M. Burgos, J. Aguilera, and J. A. Vacas, ''Design of an accurate, low-cost autonomous data logger for PV system monitoring using Arduino that complies with IEC standards,'' *Sol. Energy Mater. Sol. Cells*, vol. 130, pp. 529–543, Nov. 2014, doi: 10.1016/j.solmat.2014.08.008.

[29] A. Lopez-Vargas, M. Fuentes, M. V. Garcia, and F. J. Munoz-Rodriguez, ''Low-cost datalogger intended for remote monitoring of solar photovoltaic standalone systems based on Arduino,'' *IEEE Sensors J.*, vol. 19, no. 11, pp. 4308–4320, Jun. 2019, doi: 10.1109/JSEN.2019.2898667.

LUKE J. BRADLEY received the Bachelor of Science degree in electrical and electronic engineering from Newcastle University, in 2015, and the Doctorate of Philosophy degree in cryogenic power electronics from Newcastle University, in 2020. He is currently a Research Assistant working in subsea robotics with the Emerging Technology and Materials Department, Newcastle University. His research interests include semiconductor device and material modeling at cryogenic temperatures, power device simulations, and microcontroller data logging applications.

NICK G. WRIGHT (Member, IEEE) received the Bachelor's and Ph.D. degrees from The University of Edinburgh. He has led major projects on power semiconductor devices, robotics, and AI. He is currently a Professor in electronic materials with Newcastle University, U.K., and a Fellow with the Turing Institute. He has authored over 200 articles, contributed to over ten patents and made numerous conference presentations. His research interests include electronic materials, power devices, and the applications of materials to robotics, and artificial intelligence — particularly for manufacturing and deep ocean exploration.