Atmel AT04204: Design a Buck Converter with XMEGA E

Atmel AVR XMEGA E

Features

- Atmel® AVR® ATxmega32E5 microcontroller
- Introduce the hardware of the Buck converter
- Guide the user to calculate parameters of the Buck converter
- Guide the user to implement the Buck converter with the XMEGA® E

Description

The application note will introduce the working principle of the Buck converter, and give a guide on how to design a Buck converter with XMEGA E device in both hardware and firmware aspects. The application note also guides the user to calculate parameters of the Buck converter.
Table of Contents

1. Related items ................................................................................................................3
2. Overview .......................................................................................................................4
3. Buck Converter Parameter Calculate ...........................................................................6
   3.1 Inductor Selection ......................................................................................................6
   3.2 Output Capacitor Selection ......................................................................................7
   3.3 Freewheeling Diode Selection ................................................................................7
   3.4 Output Voltage Selection .......................................................................................8
   3.5 Input Capacitor Selection .....................................................................................9
4. The Buck Converter Implementation ...........................................................................10
   4.1 Pulse Width Modulation ......................................................................................11
   4.2 Pulse Frequency Modulation ..............................................................................12
5. Code Examples .............................................................................................................14
6. Revision History ...........................................................................................................15
1. Related items

The following list contains links to the most relevant documents for the Buck converter:

- **ATxmega32E5/ ATxmega16E5/ ATxmega8E5 Preliminary Datasheet**
  Atmel ATxmega32E5 is the microcontroller used in this solution.

- **Atmel AVR XMEGA MANUAL**
  The document contains complete and detailed description of all modules included in the Atmel AVR XMEGA E Microcontroller Family.

- **Atmel AT04022: DALI Slave with XMEGA E Hardware User Guide**
  The application note describes a Buck converter hardware for LED string with ATxmega32E5.

- **Atmel AT03922 DALI Slave with XMEGA E - Software User Guide**
  The application note describes a Buck converter software for LED string with ATxmega32E5.

- **AVR1612: PDI programming driver**
  The Program and Debug Interface (PDI) is an Atmel proprietary interface for external programming and on-chip debugging of the device. This application note describes how to implement PDI programming.

- **Atmel Studio 6**
  Atmel Studio 6 is a free Atmel IDE for development of C/C++ and assembler code for Atmel microcontrollers.

- **Atmel JTAGICE3**
  JTAGICE3 is a mid-range development tool for Atmel AVR 8- and 32-bit Microcontroller with on-chip debugging for source level symbolic debugging, Nano Trace (if supported by the device) and device programming.
2. Overview

A Buck converter is a step-down DC-to-DC converter. It converts a voltage source into a lower regulated voltage in the same polarity and is not isolated between the input and output. The input current is discontinuous due to the power switch MOSFET current that pulses from zero to a output current every switching cycle. The output current is continuous because the current is supplied by the output inductor and capacitor combination. Figure 2-1 shows the simplified diagram of the Buck converter.

Figure 2-1. Diagram of the Buck Converter

The Buck converter transfers packets of energy using a power switch MOSFET, a diode, an inductor, and several capacitors. Both N-channel and P-channel MOSFET can do the work and the user can decide which type of circuit needs be applied in their actual application. The diode D is usually called the freewheeling diode. The inductor L and capacitor C make up the output filter. The resistor RL represents the load seen by the Buck converter output. The drive circuit generates a drive signal to drive the MOSFET ON or OFF.

The Buck converter assumes two main states of operation per switching cycle, MOSFET fully ON state and MOSFET fully OFF state. The fully ON state is when Q is ON and D is cut-off. The fully OFF state is when Q is OFF and D is ON with forward current flowing through L and RL. The circuit diagram for each of the two states is shown in Figure 2-2. The orange arrows indicate the current flow.
While the MOSFET Q is fully ON, MOSFET Q is turned on, and it presents a low resistance from its drain to source, and has a small voltage drop. It also has a small voltage drop across the equal DC resistance of the inductor. The diode D is OFF during its reversed biased. The voltage applied on the right hand side of the inductor L is the output voltage. The current flows from the input power source VI, through MOSFET Q and the inductor L to the output capacitor and load resistance RL combination. The inductor L delivers current to the load. The capacitor C smoothens out the L’s current changes into a stable voltage for the load. During this time, energy is stored into the inductor in the form of a magnetic field.

Once the MOSFET is fully OFF, MOSFET Q is turned off, and it presents high impedance from its drain to source. The voltage source is isolated from the other circuit, so the output current will be dropping. The inductor is the magnetic component, and the current flowing in the inductor cannot change instantaneously. So the inductor will try to fight against the current change. In order to maintain the current when the input source is lost, the inductor will take the place of the voltage source and maintaining current flow by reversing its voltage. Although the current on the inductor is the same as the fully ON state, but the left hand side of the inductor L swings negative, it is reversed. So the freewheeling diode D turns ON and provides a path for current on the inductor to flow. The voltage applied to the right hand side of L is still the output voltage. The current flows from the ground through diode D and inductor to the output capacitor and load resistance RL combination. The inductor maintains current to the load.
3. **Buck Converter Parameter Calculate**

Below is the schematic of the Buck converter for which we will calculate component values. Figure 3-1 shows the Buck converter design schematic.

**Figure 3-1. Buck Converter Design Schematic**

![Buck Converter Design Schematic](image)

The following are key parameters calculate the Buck converter.

- Input Voltage $V_{in}$
- Output Voltage $V_{out}$
- Maximum output current $I_{load}$
- Switching frequency $f_{sw}$

The maximum Duty Cycle:

$$D = \frac{V_{out}}{V_{in}}$$

Define the Ripple current. The inductor ripple current cannot be calculated because the inductor is not known at the beginning of the design. Generally the inductor ripple current will be limited to 30% of the maximum output current.

$$I_{ripple} = 0.3 \times I_{load}$$

As a regular order, the inductor and output capacitor should be calculated first, and then determine the freewheeling diode, input capacitor and the MOSFET.

### 3.1 Inductor Selection

For the design where no inductor range is given, the following equation is a good estimation for the right inductor.

$$L = \frac{V_{out} \times (V_{in} - V_{out})}{I_{ripple} \times f_{sw} \times V_{in}}$$

After rearrange and substitute:

$$L = D \times (V_{in} - V_{out}) / (I_{ripple} \times f_{sw})$$

For the inductance, the appropriate wire diameter of the inductor should be selected. The maximum output current of the design, decides the wire diameter of the inductor. The following equation gives a reference to get the maximum output current.

$$I_{max} = I_{load} + (I_{ripple}/2)$$

To avoid the overload of the inductor, the current rating of the inductor should cover the maximum current.
As we know, a higher inductance will reduce the ripple and improve the output quality of the Buck converter. However, the higher inductance always means bigger size. It is always a tradeoff between performance and size. The designer needs to handle and balance it in their actual application.

### 3.2 Output Capacitor Selection

The voltage ripple across the output capacitor is the sum of three parts. One is due to the Effective Series Resistance (ESR) of the capacitor, other is due to the Effective Series Inductance (ESL) of the capacitor, and the third is the voltage drop due to the load current that must be supplied by the capacitor as the inductor is discharged.

The capacitor ESR value can be selected from manufacturer’s catalog. But the ESL is usually not specified by the manufacturer. So in the design, we will assume that the ESL value is zero.

**Note:** The ESL specified will become more important as the switching frequency increase.

For a capacitor:

\[
\Delta V = I_{\text{ripple}} \times \text{ESR} + (I_{\text{ripple}} \times \frac{\text{Ton}}{\text{Cout}}) + (I_{\text{ripple}} \times \frac{\text{ESL}}{\text{Ton}})
\]

\[
\text{Ton} = \frac{D}{\text{fsw}}
\]

As mentioned before, assume ESL=0, Simplify the equation:

\[
\Delta V = I_{\text{ripple}} \times \text{ESR} + (I_{\text{ripple}} \times \frac{\text{Ton}}{\text{Cout}})
\]

Rearrange:

\[
\text{Cout} = \frac{(I_{\text{ripple}} \times D/fsw)}{\Delta V - I_{\text{ripple}} \times \text{ESR}}
\]

- \(\text{Cout}\) is the minimum output capacitor
- \(I_{\text{ripple}}\) is the estimated inductor ripple current
- \(\Delta V\) is the Desired output voltage ripple
- \(D\) is Duty Cycle of the Buck converter
- \(\text{fsw}\) is minimum switching frequency of the Buck converter

The term in the rearranged equation’s denominator \((\Delta V - I_{\text{ripple}} \times \text{ESR})\) shows that the ESR rating of the capacitor is more important than the capacitor value. If the selected ESR is too large, the voltage ripple will equal or exceed the target output voltage ripple. But the denominator cannot equal to zero, it means that an infinite output capacitance is required. Thus, select a low-ESR capacitor will help to minimize the ripple on the output voltage.

### 3.3 Freewheeling Diode Selection

For the freewheeling diode, the forward current rating, and the power dissipation of the diode are key aspect to be considered.

The maximum reverse voltage on the diode is \(V_{\text{in}}\).

The average current of the diode is equal to the load current with the diode is conducting. The conducting percentage of the diode is \((1-D)\). Estimate diode current:

\[
I_D = I_{\text{load}} \times (1-D)
\]

- \(I_D\) is average forward current of the freewheeling diode
- \(D\) is Duty Cycle
- \(I_{\text{load}}\) is maximum output current
The power dissipation of the diode is equal to:

\[ P = I_D \times V_F \]

\( V_F \) is forward voltage of the diode

To reduce the switching losses, the Schottky diode is also used. With higher switching frequency, the reverse recovery time of the diode should be considered.

### 3.4 Output Voltage Selection

The output voltage of the Buck converter is decided by the feedback voltage. The resistive divider network can easily obtain the feedback voltage from the output side. It also is widely used in most Buck converters. With the given feedback voltage \( V_{fb} \) and feedback bias current \( I_{fb} \), the voltage divider can be calculated.

Figure 3-2 shows the resistive for setting the output voltage.

Figure 3-2. Resistive Divider for Setting the Output Voltage

To avoid the affection of the bias current, the current through the resistive divider needs to be at least 100 times as big as the feedback bias current:

\[ I \geq 100 \times I_{fb} \]

But too big current in will leads the extra power loss on divider resistor, a reasonable value should be selected by designer.

The resistors \( R_{up} \) and \( R_{bt} \) are calculated as below:

\[ R_{bt} = \frac{V_{fb}}{I} \]

\[ R_{up} = R_{bt} \times \left( \frac{V_{out}}{V_{fb}} - 1 \right) \]

\( R_{up}, R_{bt} \) are resistors in the resistive divider

\( V_{fb} \) is the feedback voltage, which can be found in the drive IC datasheet

\( I \) is the current through the resistive divider \( R_{up} \) and \( R_{bt} \) to GND

\( V_{out} \) is the desired output voltage.
3.5 **Input Capacitor Selection**

The same with the output capacitor just mentioned above, the input capacitor selection is primarily decided by the ESR under the requirement of the voltage ripple.

The minimum value of the input capacitor is necessary to stabilize the input voltage due to the peak current requirement of the switching power supply. The worst case ripple current on the input of the Buck converter is about one half of the load current.

Refer to Section 3.2 to get the calculation of the input capacitor.
4. **The Buck Converter Implementation**

There are two methods to implement the Buck circuit using XMEGA E device. The key difference between the two methods is the working frequency. One method uses fixed frequency and the other is with variable frequency according to the change of the load. The fixed frequency method is called Pulse Width Modulation, and the variable frequency method is called Pulse Frequency Modulation.

*Figure 4-1* shows the hardware circuit of Buck circuit with the ATxmega32E5.

*Figure 4-1. Buck Converter with the ATxmega32E5*

As marked in the circuit above, the P-CH MOSFET Q1, inductor L1, freewheeling diode D2 and capacitance C7 consist of Buck circuit. The resistance R1, R2, R3, capacitance C4, C6, and transistor Q2, Q3 consist of drive circuit for MOSFET Q1. C4 and C6 is called accelerated capacitance circuit, the value of the accelerated capacitance is usually very small. They are used to accelerate either fully on or fully off of the transistor. The capacitance C2 and C3 consist of filtering and storage circuit for Buck circuit. The resistance R4 and R5 consist of voltage feedback loop. The inductor L2, capacitance C1, C5 and MCU U1 ATxmega32E5 consist of minimum MCU system.

The Buck circuit with the XMEGA E, the switching frequency of MOSFET drive signal is up to a few hundred kHz or several MHz. So it is necessary to consider the reverse recovery time of the freewheeling diode. The reverse current on the inductor will become very large if the reverse recovery time is too long.

The voltage or current feedback loop can be used as the feedback loop. In this case, the voltage feedback loop is used. When the MCU detects the feedback voltage higher than the pre-set voltage, the MCU will immediately turn off the drive signal of the MOSFET until the feedback voltage is less than the set feedback voltage, and then turn on the drive signal of the MOSFET to turn on the MOSFET for provide power for load.

This easy implementation of Buck circuit benefits from the on-chip hardware modules – linking the Event System Controller module to the Fault Extension module in the ATxmega E. It don’t need extra firmware consumption except for configuring some Registers. The user only needs to configure the Analog Comparator Register, Fault Extension Register, Event System Register, and Timer/Counter to run a Buck circuit.

The next section will guide the user how to configure registers of the MCU in Pulse Width Modulation and Pulse Frequency Modulation.
4.1 Pulse Width Modulation

For this modulation, we only need to set a fixed switching frequency for the MCU, and the pulse width of the MOSFET’s drive will be changed according to the feedback of the feedback loop.

The example for configuring the Register:

```c
int main (void)
{
    /*Set the system clock*/
    OSC.CTRL |= OSC_RC32MEN_bm;
    while(!(OSC.STATUS & OSC_RC32MRDY_bm));
    CCP = CCP_IOREG_gc;
    CLK.CTRL = CLK_SCLKSEL_RC32M_gc;

    /*Set the I/O port*/
    PORTA.OUT = 0x00;
    PORTA.DIR = (1 << 7);
    PORTA.PIN7CTRL = PORT_OPC_PULLDOWN_gc | PORT_ISC_INPUT_DISABLE_gc;

    PORTC.OUT = (1 << 1);
    PORTC.DIR = (1 << 5) | (1 << 1);
    PORTC.PIN1CTRL = PORT_OPC_PULLDOWN_gc | PORT_ISC_INPUT_DISABLE_gc;
    PORTC.PIN5CTRL = PORT_OPC_PULLDOWN_gc | PORT_ISC_INPUT_DISABLE_gc;

    PORTD.OUT = 0x00;
    PORTD.DIR = (1 << 5) | (1 << 4) | (1 << 3);
    PORTD.PIN3CTRL = PORT_OPC_PULLDOWN_gc | PORT_ISC_INPUT_DISABLE_gc;
    PORTD.PIN4CTRL = PORT_OPC_PULLDOWN_gc | PORT_ISC_INPUT_DISABLE_gc;
    PORTD.PIN5CTRL = PORT_OPC_PULLDOWN_gc | PORT_ISC_INPUT_DISABLE_gc;

    /*Set the Timer/Counter Register*/
    PR.PRGEN = PR_RTC_bm
               | PR_XCL_bm
               | PR_EDMA_bm;
    PR.PRPA = PR_ADC_bm
              | PR_DAC_bm;
    PR.PRPC = PR_TWI_bm
              | PR_USART0_bm
              | PR_SPI_bm
              | PR_HIRES_bm
              | PR_TC4_bm;
    PR.PRPD = PR_TWI_bm
              | PR_USART0_bm
              | PR_SPI_bm
              | PR_HIRES_bm
              | PR_TC4_bm;

    TCC5.PER = 31;
    TCC5.CCB = 25;
    TCC5.CTRLB |= TC45_CCBMODE_COMP_gc;
    TCC5.CTRLB = TC45_BYTEM_NORMAL_gc
                 | TC45_CIRCEN_DISABLE_gc
                 | TC45_WGMODE_SINGLESLOPE_gc;
    TCC5.CTRL A = (0 << TC5_SYNCHEN_bp)
                  | (0 << TC5_EVSTART_bp)
                  | (0 << TC5_UPSTOP_bp)
                  | TC45_CLKSEL_DIV1_gc;
    TCC5.CTRL D = TC45_EVACT_FMODE1_gc
                  | TC45_EVSEL_CH0_gc;

    /*Set the Fault Extension Register*/
```
FAULTC5_CTRLD = FAULT_KEEPB_bm | FAULT_SRCB0_bm;

/*Set the Analog Comparator Register*/
ACA.ACMUXCTRL = AC_MUXPOS_PIN0_gc | AC_MUXNEG_SCALER_gc;
ACA.CTRLB = 18;
ACA.ACMCTRL = AC_INTMODE_RISING_gc | AC_ENABLE_bm;
/*Set the Event System Register*/
EVSYS.CH0MUX = EVSYS_CMUX_ACA_CH0_gc;

While(1)
{
}

4.2 Pulse Frequency Modulation

For this modulation, we only need to set the minimum turn-on time of the MOSFET’s drive signal, and the pulse width and switching frequency of the MOSFET’s drive signal will be changed according to the feedback of the feedback loop.

The example for configuring the registers:

```c
int main (void)
{
    /*Set the system clock*/
    OSC.CTRL |= OSC_RC32MEN_bm;
    while(!(OSC.STATUS & OSC_RC32MRDY_bm));
    CCP = CCP_IOREG_gc;
    CLK.CTRL = CLK_SCLKSEL_RC32M_gc;
    /*Set the I/O port*/
    PORTA.OUT = 0x00;
    PORTA.DIR = (1 << 7);
    PORTA.PIN7CTRL = PORT_OPC_PULLDOWN_gc | PORT_ISC_INPUT_DISABLE_gc;
    PORTC.OUT = (1 << 1);
    PORTC.DIR = (1 << 5) | (1 << 1);
    PORTC.PIN1CTRL = PORT_OPC_PULLDOWN_gc | PORT_ISC_INPUT_DISABLE_gc;
    PORTC.PIN5CTRL = PORT_OPC_PULLDOWN_gc | PORT_ISC_INPUT_DISABLE_gc;
    PORTD.OUT = 0x00;
    PORTD.DIR = (1 << 5) | (1 << 4) | (1 << 3);
    PORTD.PIN3CTRL = PORT_OPC_PULLDOWN_gc | PORT_ISC_INPUT_DISABLE_gc;
    PORTD.PIN4CTRL = PORT_OPC_PULLDOWN_gc | PORT_ISC_INPUT_DISABLE_gc;
    PORTD.PIN5CTRL = PORT_OPC_PULLDOWN_gc | PORT_ISC_INPUT_DISABLE_gc;
    /*Set the Timer/Counter Register*/
    PR.PRGEN = PR_RTC_bm | PR_XCL_bm | PR_EDMA_bm;
    PR.PRPA = PR_ADC_bm | PR_DAC_bm;
    PR.PRPC = PR_TWI_bm | PR_USART0_bm | PR_SPI_bm | PR_HIRES_bm | PR_TC4_bm;
    PR.PRPD = PR_TWI_bm | PR_USART0_bm
```
TCC5.CTRLB = TC45_BYTEM_NORMAL_gc
| TC45_CIRCE_DISABLE_gc
| TC45_WGMODE_SINGLESLOPE_gc;

TCC5.CTRLC |= TC5_POLB_bm;
TCC5.CTRL D = TC45_EVACT_FMODE2_gc
| (0 << TC5_EVDSY_bp)
| TC45_EVSEL_CH0_gc;

TCC5.CTRL E = TC45_CCDMODE_DISABLE_gc
| TC45_CCMODE_DISABLE_gc
| TC45_CBMODE_COMP_gc
| TC45_CCMODE_DISABLE_gc;

TCC5.PER = 320;
TCC5.CCB = 6;
TCC5.CTRLA |= (0 << TC5_SYNCHEN_bp)
| (0 << TC5_EVSTART_bp)
| (0 << TC5_UPSTOP_bp)
| TC45_CLKSEL_DIV1_gc;

TCC5.CTRL E = TC45_CCDMODE_DISABLE_gc
| TC45_CCMODE_DISABLE_gc
| TC45_CBMODE_COMP_gc
| TC45_CCMODE_DISABLE_gc;

/*Set the Fault Extension Register*/
FAULTC5.CTRLA = FAULT_RAMPC_RAMPA1 gc
| (0 << FAULT_FDBDB_bp)
| FAULT_PORTCTRL bmi
| (0 << FAULT_FUSE_bp)
| (0 << FAULT_FILTERE_bp)
| FAULT_SRCE_DISABLE_gc;

FAULTC5.CTRLD = (0 << FAULT_SOFTT_bp)
| FAULT_HALTB_DISABLE_g
| (1 << FAULT_RESTARTT_bp)
| (0 << FAULT_KEEPT_bp)
| FAULT_SRCB_CHN1_gc;

FAULTC5.CTRLE = (0 << FAULT_CAPTB_bp)
| (0 << FAULT_FILTERB_bp)
| (0 << FAULT_BLANKB_bp)
| (1 << FAULT_QUALB_bp;

/*Set the Analog Comparator Register*/
ACA.AC0MUXCTRL = AC_MUXPOS_PIN0_gc
| AC_MUXNEG_SCALER_gc;

ACA.CTRLA = AC_ACOUT_bm;
ACA.CTRLB = 16;
ACA.AC0CTRL = AC_INTMODE_RISING gc
| AC_INTLVL_OFF_gc
| AC_HYSMODE_SMALL_gc
| AC_ENABLE_bm;

/*Set the Event System Register*/
EVSYS.CH0MUX = EVSYS_CHMUX_ACA_CH0_gc;

PORTCFG.ACEVOUT = PORTCFG_ACOUT_PA_ga
| PORTCFG_EVOUT_PC7_gc
| PORTCFG_EVASYEN bm
| PORTCFG_EVOUTSEL_1 gc;

While(1)
{
}
}
5. **Code Examples**

The example application is based on the Atmel Software Framework included in Atmel Studio 6. The Atmel Software Framework can also be found as a separate package online at:


For more information about the code example, see the application note Atmel AT03922: DALI Slave with XMEGA E – Software User’s Guide.
6. **Revision History**

<table>
<thead>
<tr>
<th>Doc. Rev.</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>42183</td>
<td>09/2013</td>
<td>Initial document revision</td>
</tr>
</tbody>
</table>