Usage of the AVR assembly with the `lstListing` package

Dr. Benjamin Viall

August 15, 2018

1 Introduction

As of 2018 the ECE263 class uses the Atmel/Microchip ATmega328p for its asm programming. This uses a assembly language variant which is unsupported natively by the `lstlisting` package. Given that the atmel studio environment intelligently detects register names and not all registers in a give processor are available in every processor in the AVR family, syntax highlighting will always be ad hoc in LaTeX. To simplify the usage of listings in lab reports the avrListing.tex file has been released to make atmega328p code listings look "good".

2 Usage

Code listings in avr asm require only a few line in your project; include the following lines in your preamble (near your other `\usepackage` directives)

```
\usepackage[dvipsnames]{xcolor}
\usepackage{listings}
\input{avrListing.tex}
```

Now code can be included in two ways.

1. Inline listings (suitable for short code snippets)
2. File include (appendices that have full file listings)

2.1 Any listing

Regardless of the listing type you choose all asm snippets will share common markup language. Best practice suggests that snippets are shown as either figures or code listings. For the purposes of these examples, we will show all examples using the figure environment.

While it is not strictly needed to redeclare `\lstset` every time you make a figure, it will make copy/paste operations easier in the future, especially if you are using multiple languages in a document. Figure 1 demonstrates how to begin adding a figure that will contain the AVR assembly snippet. Two keyword arguments are demonstrated. the first "`language=[AVR]{Assembler}`", tells the listing package that you wish to use the keywords defined for the AVR variant of an Assembler language. The second , `style=avrasm` tells the package to use the avrasm style designed to mimic the syntax highlighting of of atmel studio. other keyword arguments can be used to customize the apperace of your source listings. and can be found in the documentation for `lstlisting`.

2.2 Inline Listings

Now that we have covered the use of any type of listing we demonstrate the use of the `lstlisting` environment. Figure 2 displays an avrAsm macro as a figure on the left and the required code to create that figure on the right. The colors

```
\begin {figure}
\label{fig:macros}
\lstset{language=[AVR]{Assembler},style=avrasm}
%paste your asm code here
\caption {a caption for this ASM listing...}
\end{figure}
```

Figure 1: Any figure that contains a asm listing.

```
;STore Imm. byte
; - Store byte to sram
;  Usage: STI [Addr16],Rt,[k8]
.macro STI ;
LDI @1,@2
STS @0,@1
.endmacro
```

Figure 2: macros...macros everywhere

```
\begin{figure}
\label{fig:macros}
\lstset{language=[AVR]{Assembler},style=avrasm}
\begin{lstlisting}[frame=none]

;STore Imm. byte
; - Store byte to sram
;  Usage: STI [Addr16],Rt,[k8]
.macro STI ;
LDI @1,@2
STS @0,@1
.endmacro
\end{lstlisting}
\caption{macros...macros everywhere}
\end{figure}
```

Figure 3: tex source for inline code listings.

are a result of the style keyword. changing the contents of the style inside of avrListing.tex is possible but not recommended. If you wish to define your own style begin by copying the style definition into your tex document and giving it a unique name, then edit to your desired color scheme.

It is also still possible to use other languages with the avrListing.tex file. In Figure 4 the differences between C and ASM listings are clear. Only the language keyword is changed. The usage of the frame keyword is used as a stylistic choice and is not needed.

2.3 Large File Listings

Finally, it is a common task to include the entire listing of your lab exercise as an appendix. To simplify this task the `listing` package adds the `lstinputlisting` command. similar to the `input` command in function it provides all the markup needed to prettify an entire file of asm. the example shown in Figure 6 demonstrates how to include a file called 'main.asm' as a figure in an appendix. Due to length the actual result of the file is not included in this document.

```
#INCLUDE <avr/io.h>
VOID main()
{
  WHILE (1)
  {
    PIND=_BV(5);
  }
}
```

Figure 4: other languages still work

```
 \begin{figure}
 \label{fig:CListingg}
 \begin{lstlisting}[language=c,
          style=avrasm,frame=single]
#include <avr/io.h>
void main()
{
  while (1)
  {
    PIND=_BV(5);
  }
}
 \end{lstlisting}
 \caption{other languages still work}
\end{figure}
```

Figure 5: Tex code for a C style Listing.

```
\begin{figure}
 \lstset{language=[AVR]{Assembler},style=avrasm}
 \lstinputlisting{main.asm}
 \caption{big file}
\end{figure}
```

Figure 6: Including an ASM file

3 Conclusion

This document barely scratches the surface of what is capable with the `listing` package. Always refer to the CTAN archive for complete documentation of any package. But this should serve as a quick reference to include avr assembly code in your own documents.