Class Documentation

digitGroup Class Reference

```
#include <digits.h>
```

Public Member Functions

- **digitGroup** (**digits** *, uint8_t, uint8_t)
  Set-up the digit group.
- void **segDisp** (uint32_t)
  Display a decimal number.
- void **segDisp** (uint32_t, uint8_t)
  Display a decimal number.
- void **segDisp** (**symType**)
  Display a Symbol or message.
- boolean **segDispSign** (int32_t)
  Display a signed number.
- boolean **segDispSign** (int32_t, uint8_t)
  Display a signed number with a decimal point.
- void **chaseAnimation** ()
  Display an animation, each call advances the position.
- void **chaseAnimation** (uint8_t)
  Display an animation at a particular position.
- void **chaseAnimation8** ()
  Display a figure eight animation, each call advances the position.
- void **chaseAnimation8** (uint8_t)
  Display an animation at a particular position.
- uint8_t **getNumDigits** ()
  Get the number of digits in the group.

Detailed Description

Interface to the digits hardware interface class for logical groups of digits.

Author:

Keegan Morrow

Version:

7 2016.08.26

Constructor & Destructor Documentation

digitGroup::digitGroup (digits * *digitsPtr*, uint8_t *offset*, uint8_t *numDigits*)

Set-up the digit group.

Parameters:

digitsPtr	Pointer to the digits object
offset	Position of the first digit in the group in the digits chain
numDigits	Number of digits in the group

Member Function Documentation

void digitGroup::chaseAnimation ()

Display an animation, each call advances the position.

void digitGroup::chaseAnimation (uint8_t *pos*)

Display an animation at a particular position.

Parameters:

pos	Position, 0-5

void digitGroup::chaseAnimation8 ()

Display a figure eight animation, each call advances the position.

void digitGroup::chaseAnimation8 (uint8_t *pos*)

Display an animation at a particular position.

Parameters:

pos	Position, 0-7

uint8_t digitGroup::getNumDigits ()

Get the number of digits in the group.

Returns:

[uint8_t] Number of digits

void digitGroup::segDisp (uint32_t *number*)

Display a decimal number.

Parameters:

number	Number to display

void digitGroup::segDisp (uint32_t *number*, uint8_t *dpPos*)

Display a decimal number.

Parameters:

number	Number to display
dpPos	Position of the decimal point (0 = none, 1 = right)

void digitGroup::segDisp (symType *sym*)

Display a Symbol or message.

Parameters:

sym	Symbol to display of type symType

boolean digitGroup::segDispSign (int32_t *number*)

Display a signed number.

Parameters:

number	Number to display

Returns:

[boolean] true if there is a sign overflow

This function will show a '-' sign on the right-most leading digit if the number is negative, if there are not enough digits, the function will return true if the sign is missing. The return value can be used to trigger a sign LED if needed.

boolean digitGroup::segDispSign (int32_t *number*, uint8_t *dpPos*)

Display a signed number with a decimal point.

Parameters:

number	Number to display
dpPos	Position of the decimal point (0 = none, 1 = right)

Returns:

[boolean] true if there is a sign overflow

This function will show a '-' sign on the right-most leading digit if the number is negative, if there are not enough digits, the function will return true if the sign is missing. The return value can be used to trigger a sign LED if needed.

The documentation for this class was generated from the following files:

- **digits.h**
- **digits.cpp**

digits Class Reference

`#include <digits.h>`

Inherits **hook**.

Public Member Functions

- **digits** (uint8_t, uint8_t, uint8_t, uint8_t)
  Sets the direction of the pins used and allocates memory.
- void **update** ()
  Send the output buffer to the chain.
- uint8_t * **getPtr** ()
  Get a pointer to the output buffer.
- uint8_t **getSize** ()
  Get the size of the output buffer in uint8_t.
- void **setDigit** (uint8_t, uint8_t, boolean)
  Display a single digit at a position on the chain.
- void **copySection** (uint8_t, uint8_t, uint8_t)
  Copy a section from one set of digits to another.

Public Attributes

- boolean **autoUpdate**

Protected Attributes

- uint8_t **numChips**
- uint8_t * **chips**

Additional Inherited Members

Detailed Description

Hardware interface class for a chain of digits.

Author:

Keegan Morrow

Version:

7 2016.08.26

Constructor & Destructor Documentation

digits::digits (uint8_t *dataPin*, uint8_t *clockPin*, uint8_t *latchPin*, uint8_t *numChips*)

Sets the direction of the pins used and allocates memory.

Parameters:

dataPin	Pin number connected to data
clockPin	Pin number connected to clock
latchPin	Pin number connected to latch
numChips	Number of total digits in the chain

Member Function Documentation

void digits::copySection (uint8_t *fromStart*, uint8_t *toStart*, uint8_t *length*)

Copy a section from one set of digits to another.

Parameters:

fromStart	Offset for the data to be copied
toStart	Offset for the destination
length	Number of digits to copy

uint8_t * digits::getPtr ()

Get a pointer to the output buffer.

Returns:

[uint8_t *] Pointer to the output buffer

uint8_t digits::getSize ()

Get the size of the output buffer in uint8_t.

Returns:

[uint8_t] Size of the output buffer

void digits::setDigit (uint8_t *digit*, uint8_t *num*, boolean *state*)

Display a single digit at a position on the chain.

Parameters:

digit	Position in the chain
num	Number to be displayed
state	State of the decimal point

void digits::update ()

Send the output buffer to the chain.

Member Data Documentation

boolean digits::autoUpdate

Determines if **digits::update()** is called automatically. Default is true.

uint8_t* digits::chips`[protected]`

Output buffer, derived classes can modify the data, but should not change the pointer address.

uint8_t digits::numChips`[protected]`

Buffer size, derived classes should not modify this.

The documentation for this class was generated from the following files:

- **digits.h**
- **digits.cpp**

hook Class Reference

`#include <hook.h>`

Inherited by **digits**.

Public Member Functions

- **hook** ()
- void **attachHook** (void(*eventHook)(void))
- void **detachHook** ()

Protected Member Functions

- void **callHook** ()

Detailed Description

Utility class providing inheritable methods to implement hooks.

Author:

Keegan Morrow

Version:

1 31.01.2014

Constructor & Destructor Documentation

hook::hook ()`[inline]`

Member Function Documentation

void hook::attachHook (void(*)(void) *eventHook*)`[inline]`

Attach the function to be called.

Parameters:

eventHook	Function pointer to the function to be attached. In the form void foo().

void hook::callHook ()`[inline]`, `[protected]`

Calls the hooked function if there is one. This should be placed in the function in the inheriting class to call the hook.

void hook::detachHook ()`[inline]`

Detach the hook.

The documentation for this class was generated from the following file:

- utility/**hook.h**

File Documentation

digits.cpp File Reference

`#include "digits.h"`

Functions

- uint8_t **_digits_iToSegs** (uint32_t inp, uint8_t *outPtr, uint8_t len, uint8_t fill)
- uint8_t **_digits_mapToSegs** (uint8_t i)

Function Documentation

uint8_t _digits_iToSegs (uint32_t *inp*, uint8_t * *outPtr*, uint8_t *len*, uint8_t *fill*)

uint8_t _digits_mapToSegs (uint8_t *i*)

digits.h File Reference

```
#include "WProgram.h"
#include <inttypes.h>
#include "utility/hook.h"
```

Classes

- class **digits**
- class **digitGroup**

Macros

- #define **DIGITS**   7

Enumerations

- enum **symType** { **blank** = 1, **err** = 2, **foul** = 3, **dash** = 4, **test** = 5 }

Macro Definition Documentation

#define DIGITS   7

Enumeration Type Documentation

enum symType

Enumerator

blank
err
foul
dash
test

utility/hook.h File Reference

Classes

- class **hook**

Macros

- #define **HOOK** 1

Macro Definition Documentation

#define HOOK 1