Space Game

Generated by Doxygen 1.8.18

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 _asteroid_t Struct Reference

```
#include <asteroid.h>
```

Data Fields

- bool isMirrored
- uint8_t timeSinceLastFrameChange
- uint8_t animationFrame

3.1.1 Detailed Description

Definition at line 19 of file asteroid.h.

3.1.2 Field Documentation

3.1.2.1 animationFrame

```
uint8_t animationFrame
```

Definition at line 22 of file asteroid.h.

Referenced by mineAsteroid().

3.1.2.2 isMirrored

```
bool isMirrored
```

Definition at line 20 of file asteroid.h.

3.1.2.3 timeSinceLastFrameChange

```
uint8_t timeSinceLastFrameChange
```

Definition at line 21 of file asteroid.h.

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

- src/objects/types/asteroid/asteroid.h

3.2 _astronaut_t Struct Reference

```
#include <astronaut.h>
```

Data Fields

- uint8_t flags
- uint8_t animationFrame
- uint8_t timeSinceLastAction

3.2.1 Detailed Description

Definition at line 27 of file astronaut.h.

3.2.2 Field Documentation

3.2.2.1 animationFrame

```
uint8_t animationFrame
```

Definition at line 29 of file astronaut.h.

3.2.2.2 flags

```
uint8_t flags
```

Definition at line 28 of file astronaut.h.

Referenced by getIsControllingSpaceship().

3.2.2.3 timeSinceLastAction

```
uint8_t timeSinceLastAction
```

Definition at line 30 of file astronaut.h.

Referenced by makeAstronautDoAction(), and moveAstronaut().

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

- src/objects/types/astronaut/astronaut.h

3.3 _background_t Struct Reference

```
#include <background.h>
```

Data Fields

- uint8_t movementState

3.3.1 Detailed Description

Definition at line 12 of file background.h.

3.3.2 Field Documentation

3.3.2.1 movementState

```
uint8_t movementState
```

Definition at line 13 of file background.h.

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

- src/objects/types/background/background.h

3.4 _bullet_t Struct Reference

```
#include <bullet.h>
```

Data Fields

- bool wereIntersectingInThePreviousFrame

3.4.1 Detailed Description

Definition at line 12 of file bullet.h.

3.4.2 Field Documentation

3.4.2.1 wereIntersectingInThePreviousFrame

```
bool wereIntersectingInThePreviousFrame
```

Definition at line 13 of file bullet.h.

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

- src/objects/types/bullet/bullet.h

3.5 _heart_t Struct Reference

```
#include <heart.h>
```

Data Fields

- uint8_t animationFrame
- uint8_t timeLived
- uint8_t timeSinceLastChange

3.5.1 Detailed Description

Definition at line 16 of file heart.h.

3.5.2 Field Documentation

3.5.2.1 animationFrame

```
uint8_t animationFrame
```

Definition at line 17 of file heart.h.

3.5.2.2 timeLived

```
uint8_t timeLived
```

Definition at line 18 of file heart.h.

3.5.2.3 timeSinceLastChange

```
uint8_t timeSinceLastChange
```

Definition at line 19 of file heart.h.

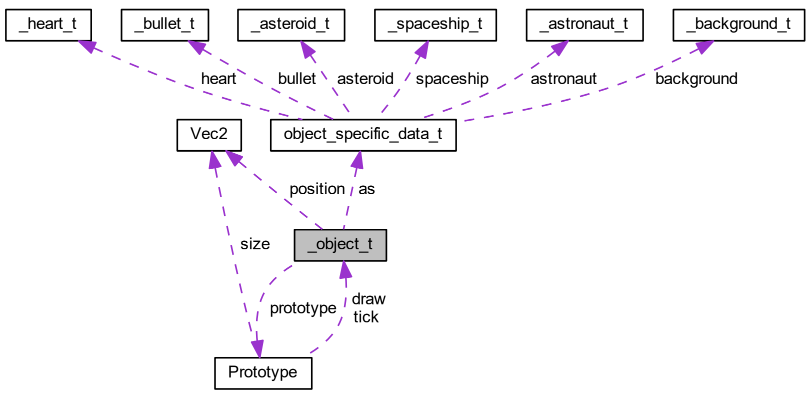
The documentation for this struct was generated from the following file:

- src/objects/types/heart/heart.h

3.6 _object_t Struct Reference

```
#include <object.h>
```

Collaboration diagram for _object_t:



Data Fields

- Prototype const ∗ prototype
- Vec2 position
- object_specific_data_t as

3.6.1 Detailed Description

Definition at line 36 of file object.h.

3.6.2 Field Documentation

3.6.2.1 as

```
object_specific_data_t as
```

Definition at line 39 of file object.h.

Referenced by getIsControllingSpaceship(), makeAstronautDoAction(), mineAsteroid(), and moveAstronaut().

3.6.2.2 position

```
Vec2 position
```

Definition at line 38 of file object.h.

Referenced by move(), and shootTurretOfSpaceship().

3.6.2.3 prototype

```
Prototype const∗ prototype
```

Definition at line 37 of file object.h.

Referenced by createObject(), drawObject(), getSize(), and tickObject().

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

- src/objects/object.h

3.7 _spaceship_t Struct Reference

```
#include <spaceship.h>
```

Data Fields

- uint8_t healthLoss
- uint8_t progress
- uint8_t activatedParts

3.7.1 Detailed Description

Definition at line 52 of file spaceship.h.

3.7.2 Field Documentation

3.7.2.1 activatedParts

```
uint8_t activatedParts
```

Definition at line 55 of file spaceship.h.

3.7.2.2 healthLoss

```
uint8_t healthLoss
```

Definition at line 53 of file spaceship.h.

3.7.2.3 progress

```
uint8_t progress
```

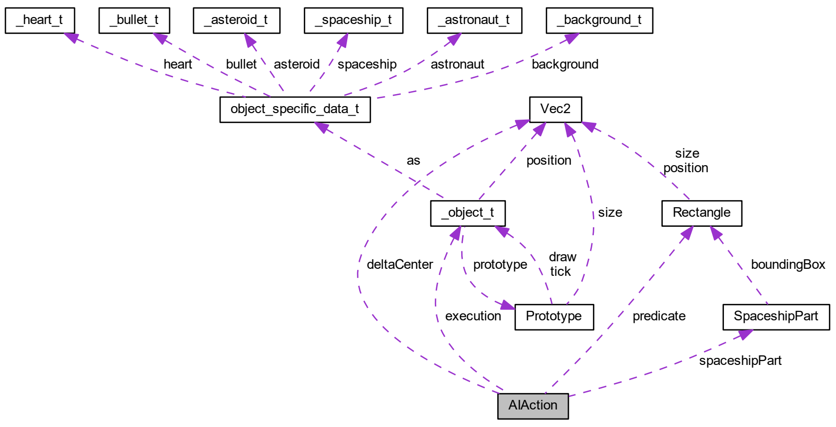
Definition at line 54 of file spaceship.h.

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

- src/objects/types/spaceship/spaceship.h

3.8 AIAction Struct Reference

Collaboration diagram for AIAction:



Data Fields

- Predicate predicate
- Execution execution
- SpaceshipPart ∗ spaceshipPart
- bool onlyOneAstronautCanDoIt
- Vec2 deltaCenter
- bool isSomeoneDoingThis

3.8.1 Detailed Description

Definition at line 22 of file ai.c.

3.8.2 Field Documentation

3.8.2.1 deltaCenter

```
Vec2 deltaCenter
```

Definition at line 27 of file ai.c.

3.8.2.2 execution

```
Execution execution
```

Definition at line 24 of file ai.c.

3.8.2.3 isSomeoneDoingThis

```
bool isSomeoneDoingThis
```

Definition at line 28 of file ai.c.

3.8.2.4 onlyOneAstronautCanDoIt

```
bool onlyOneAstronautCanDoIt
```

Definition at line 26 of file ai.c.

3.8.2.5 predicate

```
Predicate predicate
```

Definition at line 23 of file ai.c.

3.8.2.6 spaceshipPart

```
SpaceshipPart* spaceshipPart
```

Definition at line 25 of file ai.c.

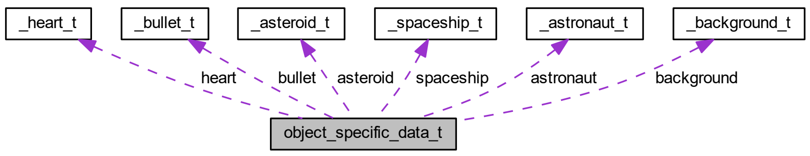
The documentation for this struct was generated from the following file:

- src/objects/ai/ai.c

3.9 object_specific_data_t Union Reference

```
#include <object.h>
```

Collaboration diagram for object_specific_data_t:



Data Fields

- struct _background_t background
- struct _spaceship_t spaceship
- struct _astronaut_t astronaut
- struct _asteroid_t asteroid
- struct _bullet_t bullet
- struct _heart_t heart

3.9.1 Detailed Description

Objects (they could have been called GameObjects) have a simple hierarchy. A prototype/flyweight motivated system is used. Each type has some common data and methods which are stored in their respective prototype. This module provides us with the methods to easily and mostly transparently access an object's prototype.

Definition at line 27 of file object.h.

3.9.2 Field Documentation

3.9.2.1 asteroid

```
struct _asteroid_t asteroid
```

Definition at line 66 of file object.h.

Referenced by mineAsteroid().

3.9.2.2 astronaut

```
struct _astronaut_t astronaut
```

Definition at line 66 of file object.h.

Referenced by getIsControllingSpaceship(), makeAstronautDoAction(), and moveAstronaut().

3.9.2.3 background

```
struct _background_t background
```

Definition at line 66 of file object.h.

3.9.2.4 bullet

```
struct _bullet_t bullet
```

Definition at line 66 of file object.h.

3.9.2.5 heart

```
struct _heart_t heart
```

Definition at line 66 of file object.h.

3.9.2.6 spaceship

```
struct _spaceship_t spaceship
```

Definition at line 66 of file object.h.

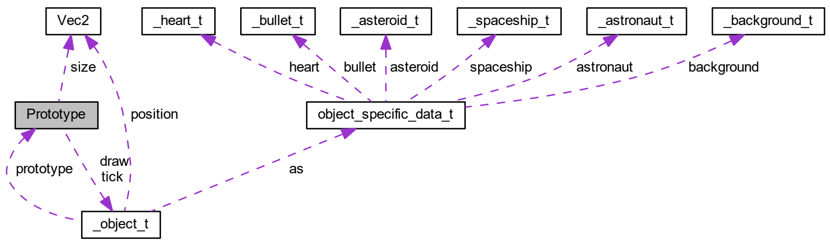
The documentation for this union was generated from the following file:

- src/objects/object.h

3.10 Prototype Struct Reference

```
#include <prototype.h>
```

Collaboration diagram for Prototype:



Data Fields

- TickMethod tick
- DrawMethod draw
- Vec2 size

3.10.1 Detailed Description

Definition at line 21 of file prototype.h.

3.10.2 Field Documentation

3.10.2.1 draw

```
DrawMethod draw
```

Definition at line 23 of file prototype.h.

Referenced by drawObject().

3.10.2.2 size

```
Vec2 size
```

Definition at line 24 of file prototype.h.

Referenced by getSizeFromPrototype().

3.10.2.3 tick

```
TickMethod tick
```

Definition at line 22 of file prototype.h.

Referenced by tickObject().

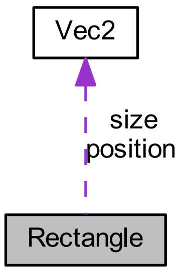
The documentation for this struct was generated from the following file:

- src/objects/prototype.h

3.11 Rectangle Struct Reference

```
#include <rectangle.h>
```

Collaboration diagram for Rectangle:



Data Fields

- Vec2 position
- Vec2 size

3.11.1 Detailed Description

Definition at line 9 of file rectangle.h.

3.11.2 Field Documentation

3.11.2.1 position

```
Vec2 position
```

Definition at line 10 of file rectangle.h.

Referenced by areIntersecting(), createObjects(), drawBitmapFromProgMem(), drawFilledRectangle(), get↩BoundingBox(), getCenter(), isBottomOnFloor(), isInside(), and translateRectangle().

3.11.2.2 size

```
Vec2 size
```

Definition at line 11 of file rectangle.h.

Referenced by areIntersecting(), getCenter(), isBottomOnFloor(), isInside(), and translateRectangle().

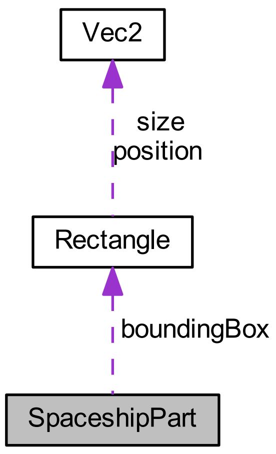
The documentation for this struct was generated from the following file:

- src/util/rectangle/rectangle.h

3.12 SpaceshipPart Struct Reference

```
#include <spaceship.h>
```

Collaboration diagram for SpaceshipPart:



Data Fields

- Rectangle boundingBox
- uint16_t ** sprite
- Action possibleAction
- bool alwaysActiveDoNotDraw

3.12.1 Detailed Description

Definition at line 34 of file spaceship.h.

3.12.2 Field Documentation

3.12.2.1 alwaysActiveDoNotDraw

```
bool alwaysActiveDoNotDraw
```

Definition at line 38 of file spaceship.h.

Referenced by isSpaceshipPartActivated().

3.12.2.2 boundingBox

```
Rectangle boundingBox
```

Definition at line 35 of file spaceship.h.

Referenced by getBoundingBoxOfSpaceshipPart().

3.12.2.3 possibleAction

```
Action possibleAction
```

Definition at line 37 of file spaceship.h.

Referenced by getPossibleActionFromSpaceship().

3.12.2.4 sprite

```
uint16_t** sprite
```

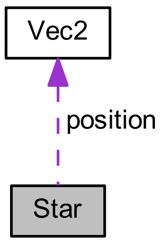
Definition at line 36 of file spaceship.h.

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

- src/objects/types/spaceship/spaceship.h

3.13 Star Struct Reference

Collaboration diagram for Star:



Data Fields

- Vec2 position
- uint8_t type

3.13.1 Detailed Description

Definition at line 13 of file background.c.

3.13.2 Field Documentation

3.13.2.1 position

```
Vec2 position
```

Definition at line 14 of file background.c.

3.13.2.2 type

```
uint8_t type
```

Definition at line 15 of file background.c.

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

- src/objects/types/background/background.c

3.14 Vec2 Struct Reference

```
#include <vec2.h>
```

Data Fields

- int8_t x
- int8_t y

3.14.1 Detailed Description

Definition at line 7 of file vec2.h.

3.14.2 Field Documentation

3.14.2.1 x

```
int8_t x
```

Definition at line 8 of file vec2.h.

Referenced by add(), areIntersecting(), clampVec2(), getCenter(), isInside(), and substract().

3.14.2.2 y

```
int8_t y
```

Definition at line 9 of file vec2.h.

Referenced by add(), areIntersecting(), clampVec2(), createObjects(), getCenter(), isBottomOnFloor(), isInside(), and substract().

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

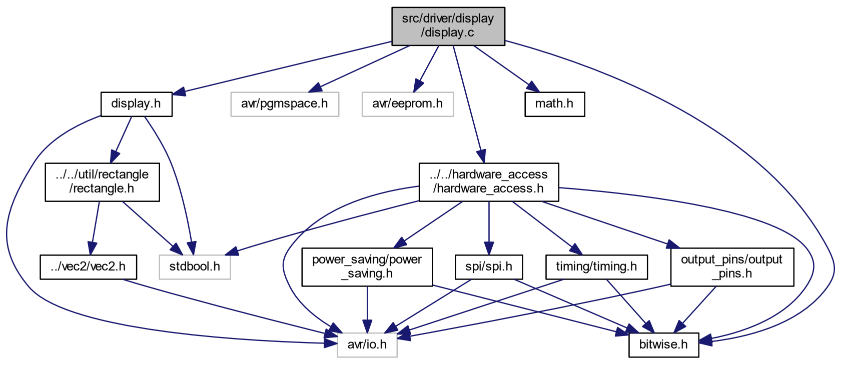
- src/util/vec2/vec2.h

Chapter 4

File Documentation

4.1 src/driver/display/display.c File Reference

```
#include "display.h"
#include <avr/pgmspace.h>
#include <avr/eeprom.h>
#include "bitwise.h"
#include "math.h"
#include "../../hardware_access/hardware_access.h"
```
Include dependency graph for display.c:



Functions

- void setDisplayContrast (uint8_t value)
- void initializeDisplay (DrawFunction drawEverything)

  Call DrawFunction n times after a call to drawFrame has been made.
- void startIntersectionTest (Rectangle compositingWindow)
- bool endIntersectionTest ()

  Return wasIntersection bit.
- void drawFrame ()

  Initiate a draw sequence.
- void drawBitmapFromProgMem (Rectangle boundingBox, uint16_t const bitmap[boundingBox.size.←
x][(boundingBox.size.y+7)/8], bool isMirrored)
- void drawFilledRectangle (Rectangle box, uint8_t invertedMask, uint8_t fill)

4.1.1 Function Documentation

4.1.1.1 drawBitmapFromProgMem()

```
void drawBitmapFromProgMem (
            Rectangle boundingBox,
            uint16_t const bitmap[boundingBox.size.x][(boundingBox.size.y+7)/8],
            bool isMirrored )
```

Draw a sprite of size boundingBox.size at boundingBox.position from bitmap if isMirrored then mirror around the vertical axis Bitmap's data is interpreted the following way: Each 16 bit word corresponds to an 8 pixel high column. (These columns are laid out horizontally from left to right. Unfortunately, the display uses this addressing mode.) The higher 8 bits of the word is the an inverted mask and the lower 8 bits are the fill bits. newPixelColumn = oldPixelColumn & ∼invertedMask | fill;

This seemingly weird layout is used to take advantage of SIMD operations and speed up the drawing process significantly.

Definition at line 91 of file display.c.

```
91
                                          {
92      boundingBox.position = substract(boundingBox.position, display.compositingWindow.position);
93
94      uint8_t spriteY = max(0, -boundingBox.position.y);
95      uint8_t spriteYByte = spriteY » 3;
96
97      for (uint8_t x = max(0, -boundingBox.position.x); x < boundingBox.size.x && boundingBox.position.x +
        x < DISPLAY_WIDTH_IN_PIXELS; x++) {
98          uint8_t spriteX = isMirrored ? boundingBox.size.x - x - 1 : x;
99          uint16_t currentPixelColumn = eeprom_read_word(&bitmap[spriteX][spriteYByte]);
100
101          uint8_t fill, invertedMask;
102          if (boundingBox.position.y >= 0) {
103              fill = (currentPixelColumn & 0x00FF) « boundingBox.position.y;
104              invertedMask = currentPixelColumn » 8 « boundingBox.position.y;
105          } else {
106              uint16_t lowerPixelColumn = spriteYByte + 1 < (boundingBox.size.y + 7) / 8 ?
        eeprom_read_word(&bitmap[spriteX][spriteYByte + 1]) : 0;
107              uint8_t shift = spriteY % 8;
108              uint8_t inverseShift = 8 - shift;
109
110              fill = ((currentPixelColumn & 0x00FF) » shift) | ((lowerPixelColumn & 0x0FF) «
        inverseShift);
111              invertedMask = (currentPixelColumn » 8 » shift) | (lowerPixelColumn » 8 « inverseShift);
112          }
113
114          compositePixelColumn(boundingBox.position.x + x, invertedMask, fill);
115      }
116 }
```

References Rectangle::position, and substract().

Here is the call graph for this function:

drawBitmapFromProgMem
substract

4.1.1.2 drawFilledRectangle()

```
void drawFilledRectangle (
            Rectangle box,
            uint8_t invertedMask,
            uint8_t fill )
```

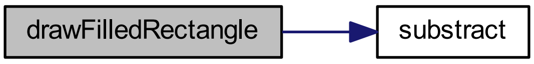
Draw a one byte repeated texture covering the parameter box for a white rectangle use these arguments: inverted←
Mask: don't care, fill: 0xFF for a black rectangle use these arguments: invertedMask: 0xFF, fill: 0x00

Definition at line 118 of file display.c.

```
                                                                  {
    box.position = substract(box.position, display.compositingWindow.position);

    uint8_t upperGapHeight = min(8, max(0, box.position.y));
    uint8_t lowerGapHeight = min(8, max(0, 8 - (box.position.y + box.size.y)));

    uint8_t actualFill = (fill » lowerGapHeight) & (fill « upperGapHeight);
    uint8_t actualInvertedMask = (invertedMask » lowerGapHeight) & (invertedMask « upperGapHeight);

    for (uint8_t x = max(0, box.position.x); x < box.position.x + box.size.x && x <
   DISPLAY_WIDTH_IN_PIXELS; x++) {
        compositePixelColumn(x, actualInvertedMask, actualFill);
    }
}
```

References Rectangle::position, and substract().

Here is the call graph for this function:



4.1.1.3 drawFrame()

```
void drawFrame ( )
```

Initiate a draw sequence.

Definition at line 71 of file display.c.

```
                 {
    display.compositingWindow = DEFAULT_COMPOSITING_WINDOW;
    for (display.compositingWindow.position.y = 0; display.compositingWindow.position.y <
    DISPLAY_HEIGHT_IN_PIXELS; display.compositingWindow.position.y += 8) {
        display.drawEverything(display.compositingWindow);

        for (uint8_t x = 0; x < DISPLAY_WIDTH_IN_PIXELS; x++) {
            sendByteOnSPI(display.compositingBuffer[x]);
            sendByteOnSPI(display.compositingBuffer[x]);
            display.compositingBuffer[x] = 0;
        }
    }
}
```

4.1.1.4 endIntersectionTest()

```
bool endIntersectionTest ( )
```

Return wasIntersection bit.

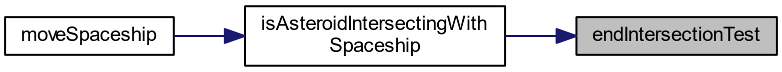
Definition at line 64 of file display.c.

```
                              {
    for (uint8_t x = 0; x < DISPLAY_WIDTH_IN_PIXELS; x++) {
        display.compositingBuffer[x] = 0;
    }
    return display.wasIntersection;
}
```

References DISPLAY_WIDTH_IN_PIXELS.

Referenced by isAsteroidIntersectingWithSpaceship().

Here is the caller graph for this function:



4.1.1.5 initializeDisplay()

```
void initializeDisplay (
            DrawFunction drawEverything )
```

Call DrawFunction n times after a call to drawFrame has been made.

some time has to elapse before the next line gets called, otherwise the display wont turn on

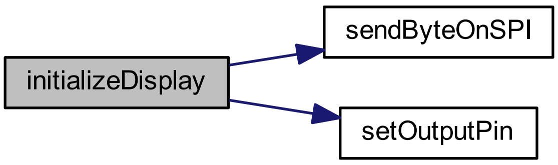
Definition at line 42 of file display.c.

```
                                                {
    setOutputPin(DISPLAY_RESET_OUTPUT_PIN, false);
    for (volatile uint8_t i = 0; i != 255; i++)
        ;
    /// some time has to elapse before the next line gets called,
    /// otherwise the display wont turn on
    setOutputPin(DISPLAY_RESET_OUTPUT_PIN, true);

    for (uint8_t i = 0; i < sizeof(configuration); i++) {
        sendByteOnSPI(pgm_read_byte(configuration + i));
    }

    display.drawEverything = drawEverything;
    setDisplayContrast(255);
    dataMode();
}
```

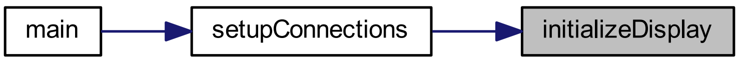
References DISPLAY_RESET_OUTPUT_PIN, sendByteOnSPI(), and setOutputPin().

Referenced by setupConnections().

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.1.6 setDisplayContrast()

```
void setDisplayContrast (
            uint8_t value )
```

Set a the brightness of the display Value can be any number between 0 and 255.

Definition at line 35 of file display.c.

```
                                          {
    commandMode();
    sendByteOnSPI(0x81);
    sendByteOnSPI(value);
    dataMode();
}
```

4.1.1.7 startIntersectionTest()

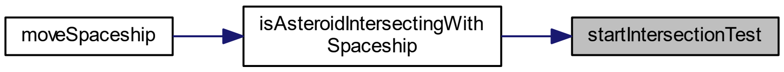
```
void startIntersectionTest (
            Rectangle compositingWindow )
```

Definition at line 59 of file display.c.

```
                                                          {
    display.wasIntersection = false;
    display.compositingWindow = compositingWindow;
}
```

Referenced by isAsteroidIntersectingWithSpaceship().

Here is the caller graph for this function:



4.1.2 Variable Documentation

4.1.2.1 compositingBuffer

```
uint8_t compositingBuffer[DISPLAY_WIDTH_IN_PIXELS]
```

Definition at line 21 of file display.c.

4.1.2.2 compositingWindow

```
Rectangle compositingWindow
```

Definition at line 22 of file display.c.

Referenced by drawObject().

4.1.2.3 drawEverything

```
DrawFunction drawEverything
```

Definition at line 23 of file display.c.

4.1.2.4 wasIntersection

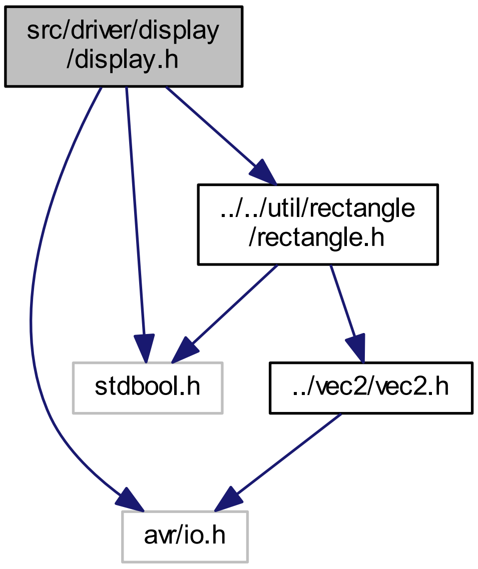
```
bool wasIntersection
```

Definition at line 24 of file display.c.

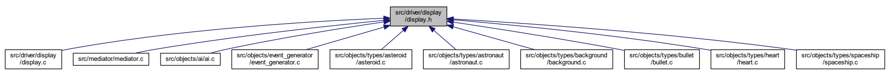
4.2 src/driver/display/display.h File Reference

```
#include <stdbool.h>
#include <avr/io.h>
#include "../../util/rectangle/rectangle.h"
```

Include dependency graph for display.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define DISPLAY_RESET_OUTPUT_PIN PB0
- #define DISPLAY_DC_OUTPUT_PIN PB4
- #define DISPLAY_WIDTH_IN_PIXELS 64

  A two-times downscaling is used for greater performance.
- #define DISPLAY_HEIGHT_IN_PIXELS 32
- #define WINDOW ((Rectangle){(Vec2){0, 0}, (Vec2){DISPLAY_WIDTH_IN_PIXELS, DISPLAY_HEIGHT_IN_PIXELS}})

  To easily access the window size.
- #define DEFAULT_COMPOSITING_WINDOW ((Rectangle){(Vec2){0, 0}, (Vec2){DISPLAY_WIDTH_IN_PIXELS, 8}})

Typedefs

- typedef void(∗ DrawFunction) (Rectangle)

Functions

- void initializeDisplay (DrawFunction drawEverything)

  Call DrawFunction n times after a call to drawFrame has been made.
- void setDisplayContrast (uint8_t value)
- void startIntersectionTest ()

  Clear buffer and wasIntersection bit.
- bool endIntersectionTest ()

  Return wasIntersection bit.
- void drawFrame ()

  Initiate a draw sequence.
- void drawBitmapFromProgMem (Rectangle boundingBox, uint16_t const bitmap[boundingBox.size.↩ x][(boundingBox.size.y+7)/8], bool isMirrored)
- void drawFilledRectangle (Rectangle box, uint8_t invertedMask, uint8_t fill)

4.2.1 Macro Definition Documentation

4.2.1.1 DEFAULT_COMPOSITING_WINDOW

```
#define DEFAULT_COMPOSITING_WINDOW ((Rectangle){(Vec2){0, 0}, (Vec2){DISPLAY_WIDTH_IN_PIXELS,
8}})
```

To conserve RAM, drawing is done in chunks of DEFAULT_COMPOSITING_WINDOW size instead of buffering the whole display and writing it out at once

Definition at line 27 of file display.h.

4.2.1.2 DISPLAY_DC_OUTPUT_PIN

```
#define DISPLAY_DC_OUTPUT_PIN PB4
```

Definition at line 16 of file display.h.

4.2.1.3 DISPLAY_HEIGHT_IN_PIXELS

```
#define DISPLAY_HEIGHT_IN_PIXELS 32
```

Definition at line 20 of file display.h.

4.2.1.4 DISPLAY_RESET_OUTPUT_PIN

```
#define DISPLAY_RESET_OUTPUT_PIN PB0
```

SPI driver for D096-12864

Definition at line 15 of file display.h.

4.2.1.5 DISPLAY_WIDTH_IN_PIXELS

```
#define DISPLAY_WIDTH_IN_PIXELS 64
```

A two-times downscaling is used for greater performance.

Definition at line 19 of file display.h.

4.2.1.6 WINDOW

```
#define WINDOW ((Rectangle){(Vec2){0, 0}, (Vec2){DISPLAY_WIDTH_IN_PIXELS, DISPLAY_HEIGHT_IN_PIXELS}})
```

To easily access the window size.

Definition at line 23 of file display.h.

4.2.2 Typedef Documentation

4.2.2.1 DrawFunction

```
typedef void(* DrawFunction) (Rectangle)
```

Definition at line 28 of file display.h.

4.2.3 Function Documentation

4.2.3.1 drawBitmapFromProgMem()

```
void drawBitmapFromProgMem (
            Rectangle boundingBox,
            uint16_t const bitmap[boundingBox.size.x][(boundingBox.size.y+7)/8],
            bool isMirrored )
```

Draw a sprite of size boundingBox.size at boundingBox.position from bitmap if isMirrored then mirror around the vertical axis Bitmap's data is interpreted the following way: Each 16 bit word corresponds to an 8 pixel high column. (These columns are laid out horizontally from left to right. Unfortunately, the display uses this addressing mode.) The higher 8 bits of the word is the an inverted mask and the lower 8 bits are the fill bits. newPixelColumn = oldPixelColumn & ∼invertedMask | fill;

This seemingly weird layout is used to take advantage of SIMD operations and speed up the drawing process significantly.

Definition at line 91 of file display.c.

```
                                        {
    boundingBox.position = substract(boundingBox.position, display.compositingWindow.position);

    uint8_t spriteY = max(0, -boundingBox.position.y);
    uint8_t spriteYByte = spriteY » 3;

    for (uint8_t x = max(0, -boundingBox.position.x); x < boundingBox.size.x && boundingBox.position.x +
    x < DISPLAY_WIDTH_IN_PIXELS; x++) {
        uint8_t spriteX = isMirrored ? boundingBox.size.x - x - 1 : x;
        uint16_t currentPixelColumn = eeprom_read_word(&bitmap[spriteX][spriteYByte]);

         uint8_t fill, invertedMask;
         if (boundingBox.position.y >= 0) {
             fill = (currentPixelColumn & 0x00FF) « boundingBox.position.y;
             invertedMask = currentPixelColumn » 8 « boundingBox.position.y;
         } else {
             uint16_t lowerPixelColumn = spriteYByte + 1 < (boundingBox.size.y + 7) / 8 ?
    eeprom_read_word(&bitmap[spriteX][spriteYByte + 1]) : 0;
             uint8_t shift = spriteY % 8;
             uint8_t inverseShift = 8 - shift;

             fill = ((currentPixelColumn & 0x00FF) » shift) | ((lowerPixelColumn & 0x0FF) «
    inverseShift);
             invertedMask = (currentPixelColumn » 8 » shift) | (lowerPixelColumn » 8 « inverseShift);
         }

         compositePixelColumn(boundingBox.position.x + x, invertedMask, fill);
     }
}
```

References Rectangle::position, and substract().

Here is the call graph for this function:

drawBitmapFromProgMem
substract

4.2.3.2 drawFilledRectangle()

```
void drawFilledRectangle (
            Rectangle box,
            uint8_t invertedMask,
            uint8_t fill )
```

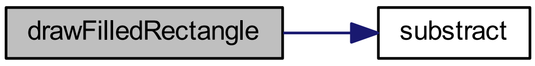
Draw a one byte repeated texture covering the parameter box for a white rectangle use these arguments: inverted←Mask: don't care, fill: 0xFF for a black rectangle use these arguments: invertedMask: 0xFF, fill: 0x00

Definition at line 118 of file display.c.

```
                                                                  {
    box.position = substract(box.position, display.compositingWindow.position);

    uint8_t upperGapHeight = min(8, max(0, box.position.y));
    uint8_t lowerGapHeight = min(8, max(0, 8 - (box.position.y + box.size.y)));

    uint8_t actualFill = (fill » lowerGapHeight) & (fill « upperGapHeight);
    uint8_t actualInvertedMask = (invertedMask » lowerGapHeight) & (invertedMask « upperGapHeight);

    for (uint8_t x = max(0, box.position.x); x < box.position.x + box.size.x && x <
   DISPLAY_WIDTH_IN_PIXELS; x++) {
        compositePixelColumn(x, actualInvertedMask, actualFill);
    }
}
```

References Rectangle::position, and substract().

Here is the call graph for this function:



4.2.3.3 drawFrame()

```
void drawFrame ( )
```

Initiate a draw sequence.

Definition at line 71 of file display.c.

```
                  {
    display.compositingWindow = DEFAULT_COMPOSITING_WINDOW;
    for (display.compositingWindow.position.y = 0; display.compositingWindow.position.y <
    DISPLAY_HEIGHT_IN_PIXELS; display.compositingWindow.position.y += 8) {
        display.drawEverything(display.compositingWindow);

        for (uint8_t x = 0; x < DISPLAY_WIDTH_IN_PIXELS; x++) {
            sendByteOnSPI(display.compositingBuffer[x]);
            sendByteOnSPI(display.compositingBuffer[x]);
            display.compositingBuffer[x] = 0;
        }
    }
}
```

4.2.3.4 endIntersectionTest()

```
bool endIntersectionTest ( )
```

Return wasIntersection bit.

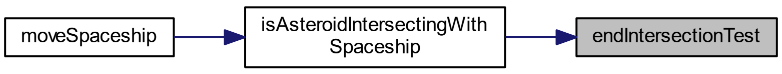
Definition at line 64 of file display.c.

```
                         {
    for (uint8_t x = 0; x < DISPLAY_WIDTH_IN_PIXELS; x++) {
        display.compositingBuffer[x] = 0;
    }
    return display.wasIntersection;
}
```

References DISPLAY_WIDTH_IN_PIXELS.

Referenced by isAsteroidIntersectingWithSpaceship().

Here is the caller graph for this function:



4.2.3.5 initializeDisplay()

```
void initializeDisplay (
            DrawFunction drawEverything )
```

Call DrawFunction n times after a call to drawFrame has been made.

some time has to elapse before the next line gets called, otherwise the display wont turn on

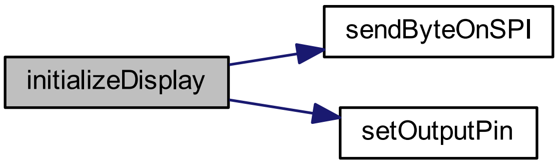
Definition at line 42 of file display.c.

```
                                                    {
    setOutputPin(DISPLAY_RESET_OUTPUT_PIN, false);
    for (volatile uint8_t i = 0; i != 255; i++)
        ;
    /// some time has to elapse before the next line gets called,
    /// otherwise the display wont turn on
    setOutputPin(DISPLAY_RESET_OUTPUT_PIN, true);

    for (uint8_t i = 0; i < sizeof(configuration); i++) {
        sendByteOnSPI(pgm_read_byte(configuration + i));
    }

    display.drawEverything = drawEverything;
    setDisplayContrast(255);
    dataMode();
}
```

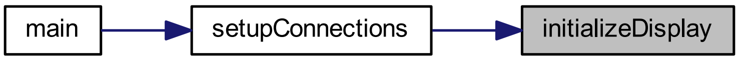
References DISPLAY_RESET_OUTPUT_PIN, sendByteOnSPI(), and setOutputPin().

Referenced by setupConnections().

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.3.6 setDisplayContrast()

```
void setDisplayContrast (
             uint8_t value )
```

Set a the brightness of the display Value can be any number between 0 and 255.

Definition at line 35 of file display.c.

```
                                    {
    commandMode();
    sendByteOnSPI(0x81);
    sendByteOnSPI(value);
    dataMode();
}
```

4.2.3.7 startIntersectionTest()

```
void startIntersectionTest ( )
```

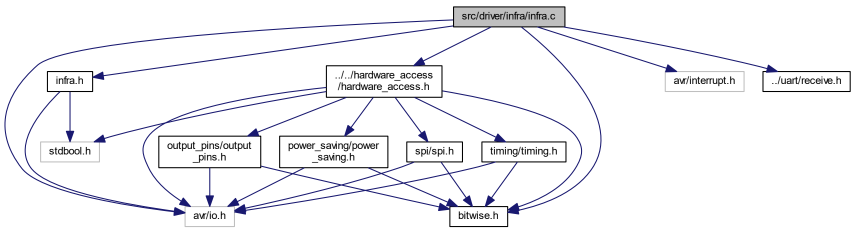
Clear buffer and wasIntersection bit.

To conserve program memory, pixel based intersection test is implemented here. Calling draw functions after calling startIntersectionTest will set a wasIntersection bit appropriately.

4.3 src/driver/infra/infra.c File Reference

```
#include "infra.h"
#include <avr/io.h>
#include <avr/interrupt.h>
#include "bitwise.h"
#include "../../hardware_access/hardware_access.h"
#include "../uart/receive.h"
```

Include dependency graph for infra.c:



Macros

- #define MEAN_OF_0_1_BIT_TIMES 53

  (0.5625 + (0.5625 + 1.6875) / 2) / 1000 / timer interval
- #define MAYBE_ONE_CHECK_TIME 141

  9 / 2 / 1000 / timer interval
- #define TIMEOUT 254

  some large value

Enumerations

- enum ProtocolState {
  idle, maybeLeadingOne, leadingOneConfirmed, activeFirstBit,
  active, timingOut }
- enum CommandState {
  noMatch, foundStartingByte, significantByte, waitingForEndOfCommand,
  waitingForRepeat }

Functions

- ISR (PCINT0_vect)
- ISR (TIM0_COMPB_vect)
- void initializeInfra (OnCommandReceived onCommandReceived)

4.3.1 Macro Definition Documentation

4.3.1.1 MAYBE_ONE_CHECK_TIME

```
#define MAYBE_ONE_CHECK_TIME 141
```

9 / 2 / 1000 / timer interval

Definition at line 15 of file infra.c.

4.3.1.2 MEAN_OF_0_1_BIT_TIMES

```
#define MEAN_OF_0_1_BIT_TIMES 53
```

(0.5625 + (0.5625 + 1.6875) / 2) / 1000 / timer interval

Definition at line 13 of file infra.c.

4.3.1.3 TIMEOUT

```
#define TIMEOUT 254
```

some large value

Definition at line 17 of file infra.c.

4.3.2 Enumeration Type Documentation

4.3.2.1 CommandState

```
enum CommandState
```

Enumerator

noMatch	
foundStartingByte	
significantByte	
waitingForEndOfCommand	
waitingForRepeat	

Definition at line 22 of file infra.c.

```
           {
    noMatch, foundStartingByte, significantByte, waitingForEndOfCommand, waitingForRepeat
} CommandState;
```

4.3.2.2 ProtocolState

```
enum ProtocolState
```

Enumerator

idle	
maybeLeadingOne	
leadingOneConfirmed	
activeFirstBit	
active	
timingOut	

Definition at line 18 of file infra.c.

```
             {
    idle, maybeLeadingOne, leadingOneConfirmed, activeFirstBit, active, timingOut
} ProtocolState;
```

4.3.3 Function Documentation

4.3.3.1 initializeInfra()

```
void initializeInfra (
            OnCommandReceived onCommandReceived )
```

Initialize infra and call onCommandReceived with every received byte Call onCommandReceived with the argument REPEAT_CODE if a repeat code has been received. enable pull-up

specific pin change interrupt enable

global on pin change interrupt enable;

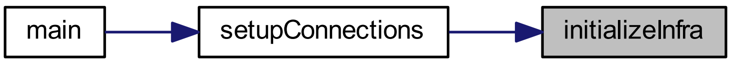
Definition at line 141 of file infra.c.

```
                                                          {
    setBit(PORTB, IR_PIN);  /// enable pull-up
    setBit(PCMSK, IR_PIN);  /// specific pin change interrupt enable
    setBit(GIMSK, PCIE);  /// global on pin change interrupt enable;
    infra.onCommandReceived = onCommandReceived;
}
```

References IR_PIN, and setBit.

Referenced by setupConnections().

Here is the caller graph for this function:



4.3.3.2 ISR() [1/2]

```
ISR (
              PCINT0_vect  )
```

Definition at line 84 of file infra.c.

```
                  {
      handlePinChangeForUartReceive();
      switch (infra.protocolState) {
          case idle:
              if (isIrOn()) {
                  enableTimerB(MAYBE_ONE_CHECK_TIME);
                  infra.protocolState = maybeLeadingOne;
              }
              break;
          case maybeLeadingOne:
              infra.protocolState = idle;
              break;
          case leadingOneConfirmed:
              if (isIrOff()) {
                  infra.protocolState = activeFirstBit;
              }
              break;
          case activeFirstBit:
          case timingOut:
              if (isIrOn()) {
                  infra.protocolState = active;
                  enableTimerB(MEAN_OF_0_1_BIT_TIMES);
              }
              break;
          case active:
              if (isIrOn()) {
                  saveBit(1);
                  enableTimerB(MEAN_OF_0_1_BIT_TIMES);
              }
              break;
      }
  }
```

References handlePinChangeForUartReceive().

Here is the call graph for this function:

ISR
handlePinChangeForUartReceive

4.3.3.3 ISR() [2/2]

```
ISR (
              TIM0_COMPB_vect  )
```

Definition at line 117 of file infra.c.

```
                      {
      switch (infra.protocolState) {
          case maybeLeadingOne:
              if (isIrOn()) {
                  infra.protocolState = leadingOneConfirmed;
              }
              disableTimerB();
```

```
            break;
        case active:
            saveBit(0);
            infra.protocolState = timingOut;
            enableTimerB(TIMEOUT);
            break;
        case timingOut:
            infra.protocolState = idle;
            saveCurrentByte();
            disableTimerB();
            break;
        default:
            disableTimerB();
            break;
    }
}
```

4.3.4 Variable Documentation

4.3.4.1 bitPosition

`uint8_t bitPosition`

Definition at line 28 of file infra.c.

4.3.4.2 commandState

`CommandState commandState`

Definition at line 30 of file infra.c.

4.3.4.3 current

`uint8_t current`

Definition at line 27 of file infra.c.

4.3.4.4 onCommandReceived

`OnCommandReceived onCommandReceived`

Definition at line 31 of file infra.c.

4.3.4.5 protocolState

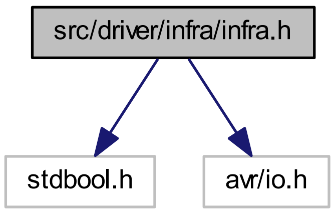
```
ProtocolState protocolState
```

Definition at line 29 of file infra.c.

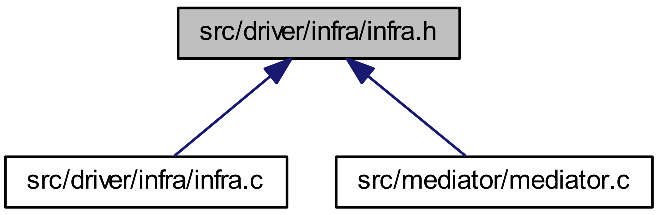
4.4 src/driver/infra/infra.h File Reference

```
#include <stdbool.h>
#include <avr/io.h>
```

Include dependency graph for infra.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define INFRA_ADDRESS 255
- #define IR_PIN PB4
- #define REPEAT_CODE 1

Typedefs

- typedef void(∗ OnCommandReceived) (uint8_t)

Functions

- void initializeInfra (OnCommandReceived onCommandReceived)

4.4.1 Macro Definition Documentation

4.4.1.1 INFRA_ADDRESS

```
#define INFRA_ADDRESS 255
```

Custom NEC implementation using a TSOP4838

Definition at line 13 of file infra.h.

4.4.1.2 IR_PIN

```
#define IR_PIN PB4
```

Definition at line 14 of file infra.h.

4.4.1.3 REPEAT_CODE

```
#define REPEAT_CODE 1
```

Definition at line 15 of file infra.h.

4.4.2 Typedef Documentation

4.4.2.1 OnCommandReceived

```
typedef void(* OnCommandReceived) (uint8_t)
```

Definition at line 16 of file infra.h.

4.4.3 Function Documentation

4.4.3.1 initializeInfra()

```
void initializeInfra (
            OnCommandReceived onCommandReceived )
```

Initialize infra and call onCommandReceived with every received byte Call onCommandReceived with the argument REPEAT_CODE if a repeat code has been received. enable pull-up

specific pin change interrupt enable

global on pin change interrupt enable;

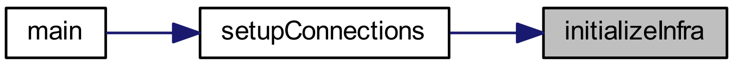
Definition at line 141 of file infra.c.

```
141                                                             {
142     setBit(PORTB, IR_PIN);  /// enable pull-up
143     setBit(PCMSK, IR_PIN);  /// specific pin change interrupt enable
144     setBit(GIMSK, PCIE);  /// global on pin change interrupt enable;
145     infra.onCommandReceived = onCommandReceived;
146 }
```

References IR_PIN, and setBit.

Referenced by setupConnections().

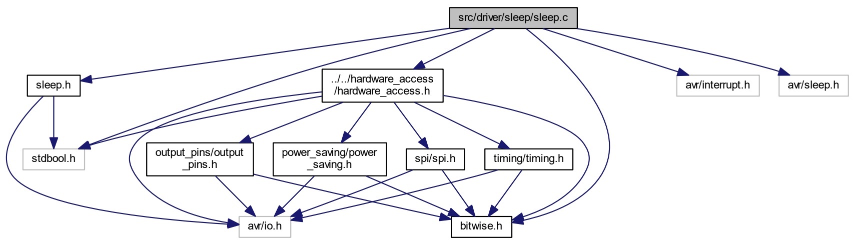
Here is the caller graph for this function:



4.5 src/driver/sleep/sleep.c File Reference

```
#include "sleep.h"
#include <stdbool.h>
#include <avr/interrupt.h>
#include <avr/sleep.h>
#include "bitwise.h"
#include "../../hardware_access/hardware_access.h"
```

Include dependency graph for sleep.c:



Macros

- #define TICKS_IN_MILISECOND 31

Functions

- void startFrameLoop (FrameFunction function, uint8_t frameLengthInMilliseconds)
  Call function every frameLengthInMilliseconds while it returns true.
- ISR (TIM0_COMPA_vect)

Variables

- volatile int8_t milisecondsSinceFrameStart

4.5.1 Macro Definition Documentation

4.5.1.1 TICKS_IN_MILISECOND

```
#define TICKS_IN_MILISECOND 31
```

Definition at line 12 of file sleep.c.

4.5.2 Function Documentation

4.5.2.1 ISR()

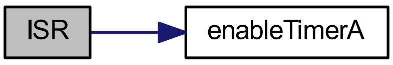
```
ISR (
                TIM0_COMPA_vect   )
```

Definition at line 31 of file sleep.c.

```
                         {
    milisecondsSinceFrameStart++;
    enableTimerA(TICKS_IN_MILISECOND);
}
```

References enableTimerA(), milisecondsSinceFrameStart, and TICKS_IN_MILISECOND.

Here is the call graph for this function:



4.5.2.2 startFrameLoop()

```
void startFrameLoop (
            FrameFunction function,
            uint8_t frameLengthInMilliseconds )
```

Call function every frameLengthInMilliseconds while it returns true.

Definition at line 15 of file sleep.c.

```
                                                                         {
      sleep_enable();

      uint8_t previousFrameTime = 0;
      while (function(previousFrameTime)) {
          previousFrameTime = milisecondsSinceFrameStart;

          while (milisecondsSinceFrameStart < frameLengthInMilliseconds) {
              sleep_cpu();
          }

          milisecondsSinceFrameStart = 0;
      }
  }
```

References milisecondsSinceFrameStart.

4.5.3 Variable Documentation

4.5.3.1 milisecondsSinceFrameStart

```
volatile int8_t milisecondsSinceFrameStart
```

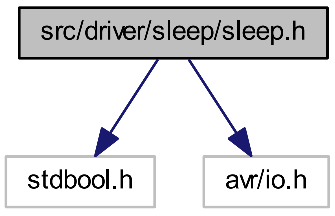
Definition at line 13 of file sleep.c.

Referenced by ISR(), and startFrameLoop().

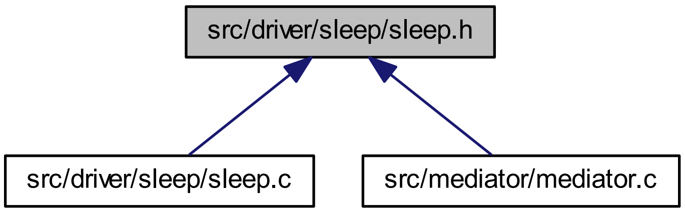
4.6 src/driver/sleep/sleep.h File Reference

```
#include <stdbool.h>
#include <avr/io.h>
```

Include dependency graph for sleep.h:



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef bool(∗ FrameFunction) (uint8_t)

  FrameFunction gets previousFrameTime (in milliseconds) as argument.

Functions

- void startFrameLoop (FrameFunction function, uint8_t frameLengthInMilliseconds)

  Call function every frameLengthInMilliseconds while it returns true.

4.6.1 Typedef Documentation

4.6.1.1 FrameFunction

```
typedef bool(* FrameFunction) (uint8_t)
```

FrameFunction gets previousFrameTime (in milliseconds) as argument.

Definition at line 8 of file sleep.h.

4.6.2 Function Documentation

4.6.2.1 startFrameLoop()

```
void startFrameLoop (
            FrameFunction function,
            uint8_t frameLengthInMilliseconds )
```

Call function every frameLengthInMilliseconds while it returns true.

Definition at line 15 of file sleep.c.

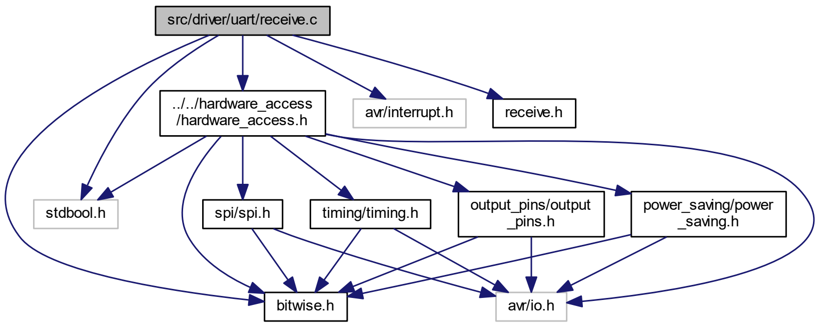
```
                                                                                {
    sleep_enable();

    uint8_t previousFrameTime = 0;
    while (function(previousFrameTime)) {
        previousFrameTime = milisecondsSinceFrameStart;

        while (milisecondsSinceFrameStart < frameLengthInMilliseconds) {
            sleep_cpu();
        }

        milisecondsSinceFrameStart = 0;
    }
}
```

References milisecondsSinceFrameStart.

4.7 src/driver/uart/receive.c File Reference

```
#include <stdbool.h>
#include <avr/interrupt.h>
#include "receive.h"
#include "bitwise.h"
#include "../../hardware_access/hardware_access.h"
```

Include dependency graph for receive.c:



Functions

- void initializeUartReceive (OnCommandReceived onCommandReceived)

  Initialize UART and call onCommandReceived with every byte received.
- void handlePinChangeForUartReceive ()

4.7.1 Function Documentation

4.7.1.1 handlePinChangeForUartReceive()

```
void handlePinChangeForUartReceive ( )
```

Check for change on the UART receive pin This function exist so we can use a single interrupt handler for both the UART receive and infra pins. Unfortunately, it's not possible to use more than one handler.

Definition at line 25 of file receive.c.

```
                                    {
    bool currentValue = getBit(PINB, RECEIVE_PIN);

    if (uartReceive.previousValue != currentValue) {
        uint8_t elapsedBitCount = (getTimeSince(uartReceive.previousTimestamp) + BIT_TIME / 2) /
    BIT_TIME;
        uartReceive.message »= elapsedBitCount;
        uartReceive.bitPosition += elapsedBitCount;

        if (uartReceive.bitPosition == 9) {
            uartReceive.onCommandReceived((uint8_t)(uartReceive.message));
            uartReceive.bitPosition = 0;
        }

        if (uartReceive.previousValue == true && currentValue == false) {
            if (uartReceive.bitPosition != elapsedBitCount) {
                uartReceive.message |= (uint8_t)(0xFF « (8 - elapsedBitCount));
            } else {
                uartReceive.bitPosition = 0;
            }
        }

        uartReceive.previousTimestamp = getTimestampFromFastTimer();
        uartReceive.previousValue = currentValue;
    }
}
```

Referenced by ISR().

Here is the caller graph for this function:

ISR
handlePinChangeForUartReceive

4.7.1.2 initializeUartReceive()

```
void initializeUartReceive (
            OnCommandReceived onCommandReceived )
```

Initialize UART and call onCommandReceived with every byte received.

setup pull-up

enable interrupt on pin

Definition at line 19 of file receive.c.

```
                                                          {
      setBit(PORTB, RECEIVE_PIN);        /// setup pull-up
      setBit(PCMSK, RECEIVE_PIN);        /// enable interrupt on pin
      uartReceive.onCommandReceived = onCommandReceived;
  }
```

References RECEIVE_PIN, and setBit.

Referenced by setupConnections().

Here is the caller graph for this function:

main
setupConnections
initializeUartReceive

4.7.2 Variable Documentation

4.7.2.1 bitPosition

```
uint8_t bitPosition
```

Definition at line 11 of file receive.c.

4.7.2.2 message

```
uint16_t message
```

Definition at line 10 of file receive.c.

4.7.2.3 onCommandReceived

```
OnCommandReceived onCommandReceived
```

Definition at line 14 of file receive.c.

4.7.2.4 previousTimestamp

```
uint8_t previousTimestamp
```

Definition at line 13 of file receive.c.

4.7.2.5 previousValue

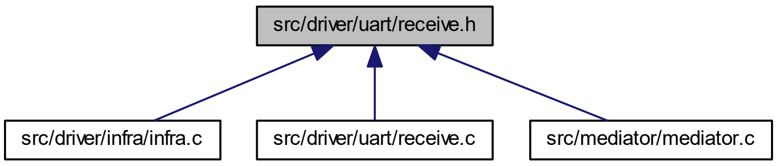
```
bool previousValue
```

goes from 0 to 9

Definition at line 12 of file receive.c.

4.8 src/driver/uart/receive.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define BIT_TIME 26

  8∗10^6 / 128 / 2400
- #define RECEIVE_PIN PB3

Typedefs

- typedef void(∗ OnCommandReceived) (uint8_t)

Functions

- void initializeUartReceive (OnCommandReceived onCommandReceived)

  Initialize UART and call onCommandReceived with every byte received.
- void handlePinChangeForUartReceive ()

4.8.1 Macro Definition Documentation

4.8.1.1 BIT_TIME

```
#define BIT_TIME 26
```

$8*10^6$ / 128 / 2400

Custom UART receive implementation with a baud rate of 2400

Definition at line 11 of file receive.h.

4.8.1.2 RECEIVE_PIN

```
#define RECEIVE_PIN PB3
```

Definition at line 12 of file receive.h.

4.8.2 Typedef Documentation

4.8.2.1 OnCommandReceived

```
typedef void(* OnCommandReceived) (uint8_t)
```

Definition at line 13 of file receive.h.

4.8.3 Function Documentation

4.8.3.1 handlePinChangeForUartReceive()

```
void handlePinChangeForUartReceive ( )
```

Check for change on the UART receive pin This function exist so we can use a single interrupt handler for both the UART receive and infra pins. Unfortunately, it's not possible to use more than one handler.

Definition at line 25 of file receive.c.

```
                                    {
    bool currentValue = getBit(PINB, RECEIVE_PIN);

    if (uartReceive.previousValue != currentValue) {
        uint8_t elapsedBitCount = (getTimeSince(uartReceive.previousTimestamp) + BIT_TIME / 2) /
    BIT_TIME;
        uartReceive.message »= elapsedBitCount;
        uartReceive.bitPosition += elapsedBitCount;

        if (uartReceive.bitPosition == 9) {
            uartReceive.onCommandReceived((uint8_t)(uartReceive.message));
            uartReceive.bitPosition = 0;
        }

        if (uartReceive.previousValue == true && currentValue == false) {
            if (uartReceive.bitPosition != elapsedBitCount) {
                uartReceive.message |= (uint8_t)(0xFF « (8 - elapsedBitCount));
            } else {
                uartReceive.bitPosition = 0;
            }
        }

        uartReceive.previousTimestamp = getTimestampFromFastTimer();
        uartReceive.previousValue = currentValue;
    }
}
```

Referenced by ISR().

Here is the caller graph for this function:

ISR
handlePinChangeForUartReceive

4.8.3.2 initializeUartReceive()

```
void initializeUartReceive (
            OnCommandReceived onCommandReceived )
```

Initialize UART and call onCommandReceived with every byte received.

setup pull-up

enable interrupt on pin

Definition at line 19 of file receive.c.

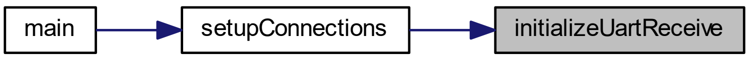
```
                                                              {
```

```
    setBit(PORTB, RECEIVE_PIN);         /// setup pull-up
    setBit(PCMSK, RECEIVE_PIN);         /// enable interrupt on pin
    uartReceive.onCommandReceived = onCommandReceived;
}
```

References RECEIVE_PIN, and setBit.

Referenced by setupConnections().

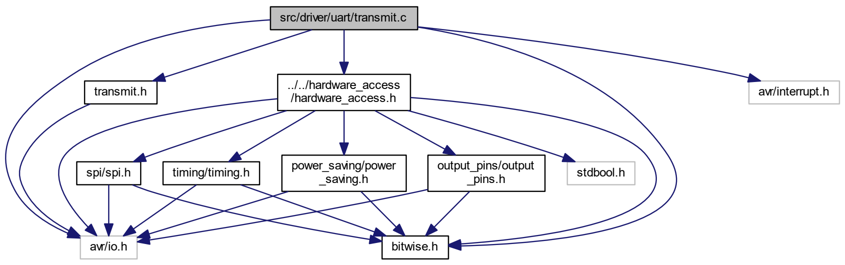
Here is the caller graph for this function:



4.9 src/driver/uart/transmit.c File Reference

```
#include "transmit.h"
#include <avr/io.h>
#include <avr/interrupt.h>
#include "bitwise.h"
#include "../../hardware_access/hardware_access.h"
```

Include dependency graph for transmit.c:



Functions

- ISR (TIM1_COMPA_vect)
- void sendByteOnUartAsync (char byte)
- void sendTextOnUartAsync (char text[ ])
- void sendUintOnUartAsync (uint8_t number)

    Send the decimal form of unsigned integer over UART.
- void initializeUartTransmit ()

    Start pulling-up the transmit line.

4.9.1 Function Documentation

4.9.1.1 initializeUartTransmit()

```
void initializeUartTransmit ( )
```

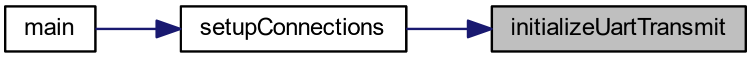
Start pulling-up the transmit line.

Definition at line 72 of file transmit.c.

```
                                 {
    send1();
}
```

Referenced by setupConnections().

Here is the caller graph for this function:



4.9.1.2 ISR()

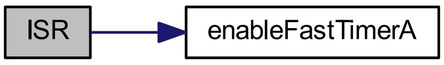
```
ISR (
            TIM1_COMPA_vect  )
```

Definition at line 25 of file transmit.c.

```
                       {
    enableFastTimerA(BIT_TIME);
    switch(uartTransmitter.currentMaskPosition) {
        case 0:
            send0();
            uartTransmitter.currentMaskPosition++;
            break;
        case 9:
            send1();
            uartTransmitter.start = (uartTransmitter.start + 1) % ASYNC_BUFFER_SIZE;
            if (uartTransmitter.start == uartTransmitter.end) {
                disableFastTimerA();
            }
            uartTransmitter.currentMaskPosition = 0;
            break;
        default:
            uartTransmitter.bytes[uartTransmitter.start] & BV(uartTransmitter.currentMaskPosition - 1) ?
    send1() : send0();
            uartTransmitter.currentMaskPosition++;
    }
}
```

References BIT_TIME, and enableFastTimerA().

Here is the call graph for this function:



4.9.1.3 sendByteOnUartAsync()

```
void sendByteOnUartAsync (
            char byte )
```

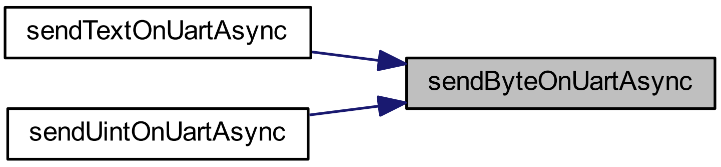
Send a single byte over on UART It's done in an asynchronous fashion using a circular buffer.

Definition at line 46 of file transmit.c.

```
                                      {
    cli();
    uartTransmitter.bytes[uartTransmitter.end] = byte;
    if (uartTransmitter.start == uartTransmitter.end) {
        enableFastTimerA(BIT_TIME);
    }
    uartTransmitter.end = (uartTransmitter.end + 1) % ASYNC_BUFFER_SIZE;

    sei();
}
```

Referenced by sendTextOnUartAsync(), and sendUintOnUartAsync().

Here is the caller graph for this function:



4.9.1.4 sendTextOnUartAsync()

```
void sendTextOnUartAsync (
            char text[] )
```

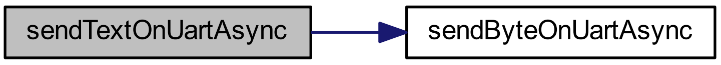
Send a string over on UART It's done in an asynchronous fashion using a circular buffer.

Definition at line 57 of file transmit.c.

```
                                        {
    for (uint8_t i = 0; text[i]; i++) {
        sendByteOnUartAsync(text[i]);
    }
}
```

References sendByteOnUartAsync().

Here is the call graph for this function:



4.9.1.5 sendUintOnUartAsync()

```
void sendUintOnUartAsync (
            uint8_t number )
```

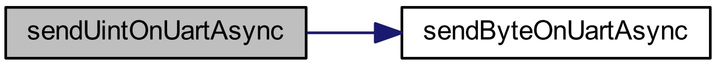
Send the decimal form of unsigned integer over UART.

Definition at line 63 of file transmit.c.

```
                                           {
    sendByteOnUartAsync(number >= 100 ? '0' + number / 100 : ' ');
    number %= 100;
    sendByteOnUartAsync(number >= 10 ? '0' + number / 10 : ' ');
    number %= 10;
    sendByteOnUartAsync('0' + number);
}
```

References sendByteOnUartAsync().

Here is the call graph for this function:



4.9.2 Variable Documentation

4.9.2.1 bytes

```
char bytes[ASYNC_BUFFER_SIZE]
```

Definition at line 10 of file transmit.c.

4.9.2.2 currentMaskPosition

```
uint8_t currentMaskPosition
```

0 - start bit, 1-8 word bits, 9 - end bit

Definition at line 14 of file transmit.c.

4.9.2.3 end

```
uint8_t end
```

Definition at line 12 of file transmit.c.

4.9.2.4 start

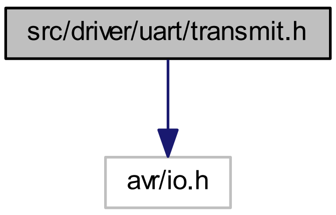
```
uint8_t start
```

Definition at line 11 of file transmit.c.

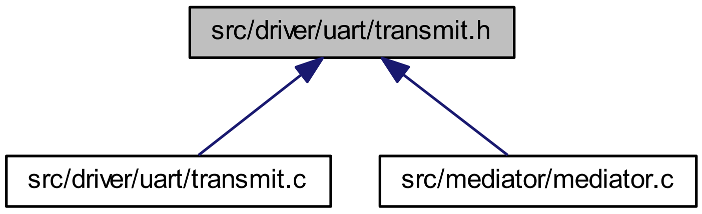
4.10 src/driver/uart/transmit.h File Reference

`#include <avr/io.h>`

Include dependency graph for transmit.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define BIT_TIME 26

  *8$*$10^6 / 128 / 2400*
- #define ASYNC_BUFFER_SIZE 20

  Messages cannot be longer than this many characters (including trailing zero)
- #define UART_OUTPUT_PIN 2

Functions

- void initializeUartTransmit ()

  Start pulling-up the transmit line.
- void sendByteOnUartAsync (char byte)
- void sendTextOnUartAsync (char text[ ])
- void sendUintOnUartAsync (uint8_t number)

  Send the decimal form of unsigned integer over UART.

4.10.1 Macro Definition Documentation

4.10.1.1 ASYNC_BUFFER_SIZE

```
#define ASYNC_BUFFER_SIZE 20
```

Messages cannot be longer than this many characters (including trailing zero)

Definition at line 16 of file transmit.h.

4.10.1.2 BIT_TIME

```
#define BIT_TIME 26
```

$8*10^6$ / 128 / 2400

Custom UART trasnmit implementation with a baud rate of 2400

Definition at line 13 of file transmit.h.

4.10.1.3 UART_OUTPUT_PIN

```
#define UART_OUTPUT_PIN 2
```

Definition at line 18 of file transmit.h.

4.10.2 Function Documentation

4.10.2.1 initializeUartTransmit()

```
void initializeUartTransmit ( )
```

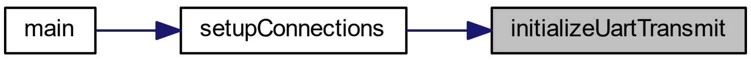
Start pulling-up the transmit line.

Definition at line 72 of file transmit.c.

```
72                              {
73     send1();
74 }
```

Referenced by setupConnections().

Here is the caller graph for this function:



4.10.2.2 sendByteOnUartAsync()

```
void sendByteOnUartAsync (
            char byte )
```

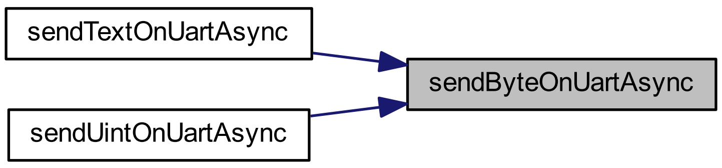
Send a single byte over on UART It's done in an asynchronous fashion using a circular buffer.

Definition at line 46 of file transmit.c.

```
                                 {
    cli();
    uartTransmitter.bytes[uartTransmitter.end] = byte;
    if (uartTransmitter.start == uartTransmitter.end) {
        enableFastTimerA(BIT_TIME);
    }
    uartTransmitter.end = (uartTransmitter.end + 1) % ASYNC_BUFFER_SIZE;

    sei();
}
```

Referenced by sendTextOnUartAsync(), and sendUintOnUartAsync().

Here is the caller graph for this function:



4.10.2.3 sendTextOnUartAsync()

```
void sendTextOnUartAsync (
            char text[] )
```

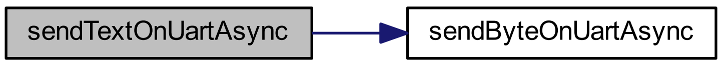
Send a string over on UART It's done in an asynchronous fashion using a circular buffer.

Definition at line 57 of file transmit.c.

```
                                   {
    for (uint8_t i = 0; text[i]; i++) {
        sendByteOnUartAsync(text[i]);
    }
}
```

References sendByteOnUartAsync().

Here is the call graph for this function:



4.10.2.4 sendUintOnUartAsync()

```
void sendUintOnUartAsync (
              uint8_t number )
```

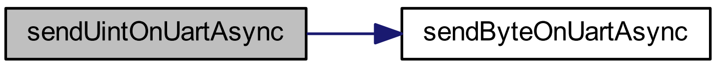
Send the decimal form of unsigned integer over UART.

Definition at line 63 of file transmit.c.

```
                                              {
       sendByteOnUartAsync(number >= 100 ? '0' + number / 100 : ' ');
       number %= 100;
       sendByteOnUartAsync(number >= 10 ? '0' + number / 10 : ' ');
       number %= 10;
       sendByteOnUartAsync('0' + number);
}
```

References sendByteOnUartAsync().

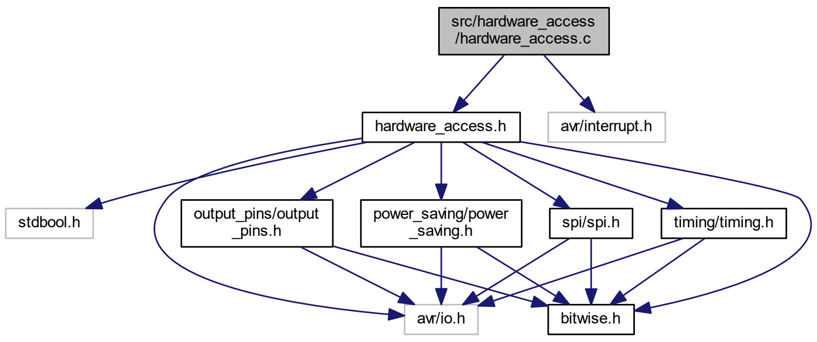
Here is the call graph for this function:



4.11 src/hardware_access/hardware_access.c File Reference

```
#include "hardware_access.h"
#include <avr/interrupt.h>
```

Include dependency graph for hardware_access.c:



Functions

- void initializeHardwareAccess ()

    Initialize every hardware element at once.

4.11.1 Function Documentation

4.11.1.1 initializeHardwareAccess()

```
void initializeHardwareAccess ( )
```

Initialize every hardware element at once.

This module contains the lowest level functions to manipulate the hardware. You only have to include this header file which serves as a facade. The sub-modules' implementation can be freely changed as long as they still implement these functions.

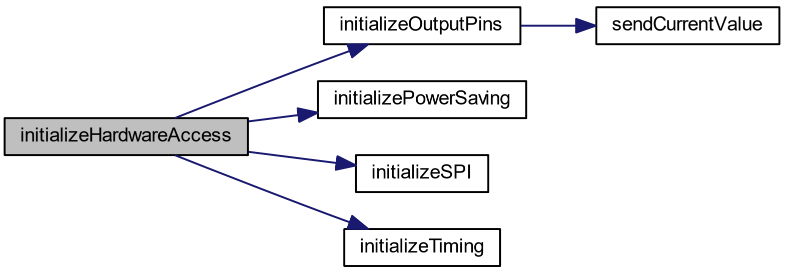
Definition at line 6 of file hardware_access.c.

```
6                                 {
7     sei();
8     initializePowerSaving();
9     initializeSPI();
10     initializeTiming();
11     initializeOutputPins();
12 }
```

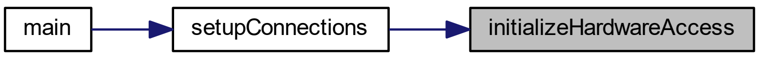
References initializeOutputPins(), initializePowerSaving(), initializeSPI(), and initializeTiming().

Referenced by setupConnections().

Here is the call graph for this function:



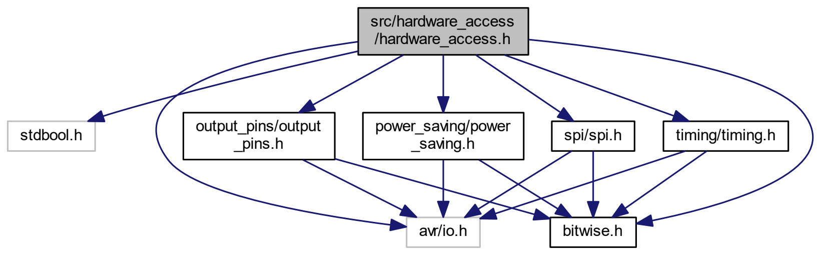
Here is the caller graph for this function:



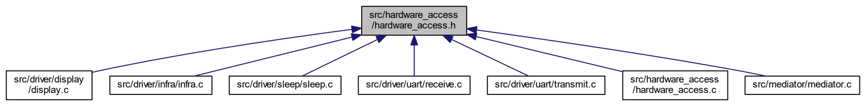
4.12 src/hardware_access/hardware_access.h File Reference

```
#include <stdbool.h>
#include <avr/io.h>
#include "bitwise.h"
#include "power_saving/power_saving.h"
#include "spi/spi.h"
#include "timing/timing.h"
#include "output_pins/output_pins.h"
```

Include dependency graph for hardware_access.h:



This graph shows which files directly or indirectly include this file:



Functions

- void initializeHardwareAccess ()

    Initialize every hardware element at once.
- void enableTimerA (uint8_t triggerInterruptInXTicks)

    Enable interrupt OCCRA for TIMER0 with a modulo of triggerInterruptInXTicks.
- void enableTimerB (uint8_t triggerInterruptInXTicks)

    Enable interrupt OCCRB for TIMER0B with a modulo of triggerInterruptInXTicks.
- void disableTimerB ()
- void enableFastTimerA (uint8_t triggerInterruptInXTicks)

    Enable interrupt OCCRA for TIMER1 with a modulo of triggerInterruptInXTicks.
- void disableFastTimerA ()
- uint8_t getTimestampFromFastTimer ()

    Return TCNT1.
- uint8_t getTimeSince (uint8_t timestamp)
- void sendByteOnSPI (uint8_t byte)
- void setOutputPin (uint8_t id, bool value)

    Set the value of an output pin.

4.12.1 Function Documentation

4.12.1.1 disableFastTimerA()

```
void disableFastTimerA ( )
```

Definition at line 25 of file timing.c.

```
                          {
    clearBit(TIMSK, OCIE1A);
}
```

References clearBit.

4.12.1.2 disableTimerB()

```
void disableTimerB ( )
```

Definition at line 15 of file timing.c.

```
                      {
    clearBit(TIMSK, OCIE0B);
}
```

References clearBit.

4.12.1.3 enableFastTimerA()

```
void enableFastTimerA (
             uint8_t triggerInterruptInXTicks )
```

Enable interrupt OCCRA for TIMER1 with a modulo of triggerInterruptInXTicks.

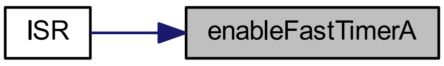
Definition at line 19 of file timing.c.

```
                                                        {
    setBit(TIFR, OCF1A);
    OCR1A = TCNT1 + triggerInterruptInXTicks;
    setBit(TIMSK, OCIE1A);
}
```

References setBit.

Referenced by ISR().

Here is the caller graph for this function:



4.12.1.4 enableTimerA()

```
void enableTimerA (
            uint8_t triggerInterruptInXTicks )
```

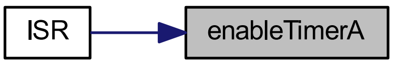
Enable interrupt OCCRA for TIMER0 with a modulo of triggerInterruptInXTicks.

Definition at line 5 of file timing.c.

```
                                              {
    OCR0A = TCNT0 + triggerInterruptInXTicks;
}
```

Referenced by ISR().

Here is the caller graph for this function:



4.12.1.5 enableTimerB()

```
void enableTimerB (
            uint8_t triggerInterruptInXTicks )
```

Enable interrupt OCCRB for TIMER0B with a modulo of triggerInterruptInXTicks.

Definition at line 9 of file timing.c.

```
                                              {
    setBit(TIFR, OCF0B);
    OCR0B = TCNT0 + triggerInterruptInXTicks;
    setBit(TIMSK, OCIE0B);
}
```

References setBit.

4.12.1.6 getTimeSince()

```
uint8_t getTimeSince (
            uint8_t timestamp )
```

Return the time since a timestamp returned by getTimestampFromFastTimer Accounts for overflow.

Definition at line 33 of file timing.c.

```
                                     {
    return timestamp <= TCNT1 ?
        TCNT1 - timestamp :
        (uint8_t)(255 - timestamp) + TCNT1;
}
```

4.12.1.7 getTimestampFromFastTimer()

```
uint8_t getTimestampFromFastTimer ( )
```

Return TCNT1.

Definition at line 29 of file timing.c.

```
                                  {
    return TCNT1;
}
```

4.12.1.8 initializeHardwareAccess()

```
void initializeHardwareAccess ( )
```

Initialize every hardware element at once.

This module contains the lowest level functions to manipulate the hardware. You only have to include this header file which serves as a facade. The sub-modules' implementation can be freely changed as long as they still implement these functions.

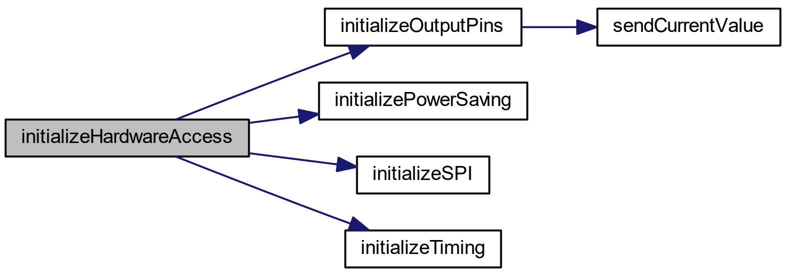
Definition at line 6 of file hardware_access.c.

```
                                 {
    sei();
    initializePowerSaving();
    initializeSPI();
    initializeTiming();
    initializeOutputPins();
}
```

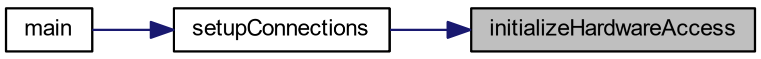
References initializeOutputPins(), initializePowerSaving(), initializeSPI(), and initializeTiming().

Referenced by setupConnections().

Here is the call graph for this function:



Here is the caller graph for this function:



4.12.1.9 sendByteOnSPI()

```
void sendByteOnSPI (
             uint8_t byte )
```

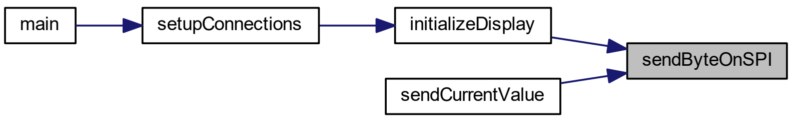
Send a single byte on the built-in SPI interface The transfer is done in a serial manner to achieve greater throughput.

Definition at line 5 of file spi.c.

```
                                {
     cli();
     USIDR = byte;
     for (uint8_t i = 16; i != 0; i--) {
         USICR = BV(USIWM0) | BV(USICLK) | BV(USITC) | BV(USICS1);
      }
      sei();
 }
```

Referenced by initializeDisplay(), and sendCurrentValue().

Here is the caller graph for this function:



4.12.1.10 setOutputPin()

```
void setOutputPin (
             uint8_t id,
             bool value )
```

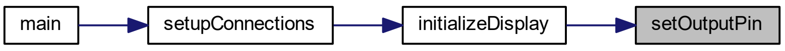
Set the value of an output pin.

Definition at line 16 of file output_pins.c.

```
                                       {
      currentValue = (currentValue & ~BV(id)) | (value & 1) « id;
      sendCurrentValue();
 }
```

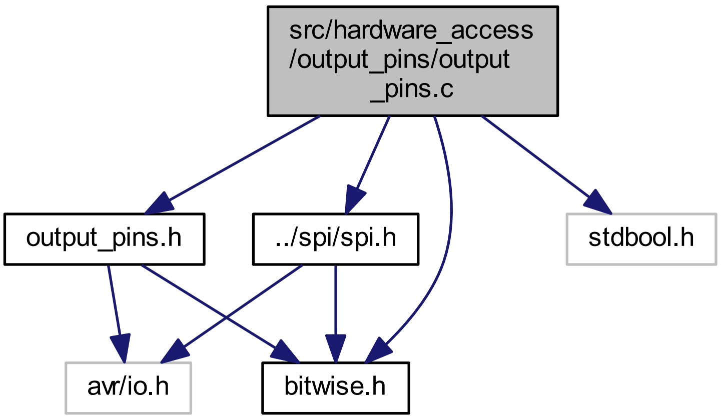
Referenced by initializeDisplay().

Here is the caller graph for this function:



4.13 src/hardware_access/output_pins/output_pins.c File Reference

```
#include "output_pins.h"
#include "bitwise.h"
#include "../spi/spi.h"
#include <stdbool.h>
```
Include dependency graph for output_pins.c:



Functions

- void sendCurrentValue ()

  Uses a 74HC595 to extend the number of digital outputs.
- void setOutputPin (uint8_t id, bool value)

  Set the value of an output pin.

4.13.1 Function Documentation

4.13.1.1 sendCurrentValue()

```
void sendCurrentValue ( )
```

Uses a 74HC595 to extend the number of digital outputs.

Definition at line 10 of file output_pins.c.
```
10                          {
11     clearBit(PORTB, CLK_ST_PIN);
12     sendByteOnSPI(currentValue);
13     setBit(PORTB, CLK_ST_PIN);
14 }
```

4.13.1.2 setOutputPin()

```
void setOutputPin (
            uint8_t id,
            bool value )
```

Set the value of an output pin.

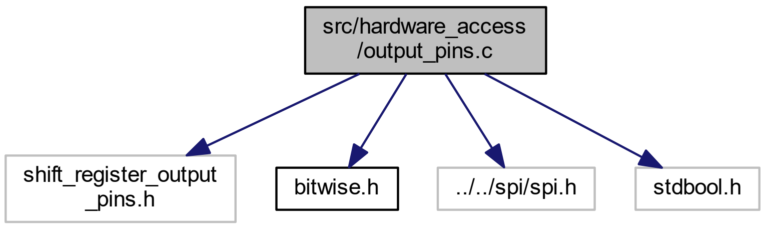
Definition at line 16 of file output_pins.c.

```
                                          {
      currentValue = (currentValue & ~BV(id)) | (value & 1) « id;
      sendCurrentValue();
}
```

4.14 src/hardware_access/output_pins.c File Reference

```
#include "shift_register_output_pins.h"
#include "bitwise.h"
#include "../../spi/spi.h"
#include <stdbool.h>
```

Include dependency graph for output_pins.c:



Functions

- void sendCurrentValue ()

    Uses a 74HC595 to extend the number of digital outputs.
- void setOutputPin (uint8_t id, bool value)

    Set the value of an output pin.

4.14.1 Function Documentation

4.14.1.1 sendCurrentValue()

```
void sendCurrentValue ( )
```

Uses a 74HC595 to extend the number of digital outputs.

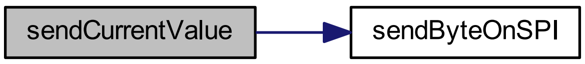
Definition at line 10 of file output_pins.c.

```
                          {
    clearBit(PORTB, CLK_ST_PIN);
    sendByteOnSPI(currentValue);
    setBit(PORTB, CLK_ST_PIN);
}
```

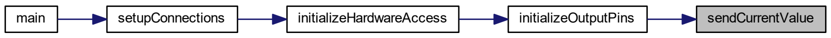
References clearBit, CLK_ST_PIN, and sendByteOnSPI().

Referenced by initializeOutputPins().

Here is the call graph for this function:



Here is the caller graph for this function:



4.14.1.2 setOutputPin()

```
void setOutputPin (
            uint8_t id,
            bool value )
```

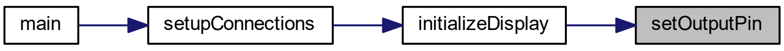
Set the value of an output pin.

Definition at line 16 of file output_pins.c.

```
                                            {
    currentValue = (currentValue & ~BV(id)) | (value & 1) « id;
    sendCurrentValue();
}
```

Referenced by initializeDisplay().

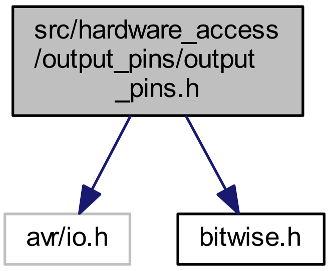
Here is the caller graph for this function:



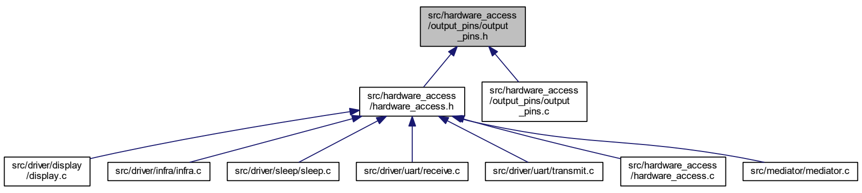
4.15 src/hardware_access/output_pins/output_pins.h File Reference

```
#include <avr/io.h>
#include "bitwise.h"
```

Include dependency graph for output_pins.h:



This graph shows which files directly or indirectly include this file:


src/hardware_access/output_pins/output_pins.h
src/hardware_access/hardware_access.h
src/hardware_access/output_pins/output_pins.c
src/driver/display/display.c
src/driver/infra/infra.c
src/driver/sleep/sleep.c
src/driver/uart/receive.c
src/driver/uart/transmit.c
src/hardware_access/hardware_access.c
src/mediator/mediator.c

Macros

- #define CLK_ST_PIN PB0

Functions

- void sendCurrentValue ()

  Uses a 74HC595 to extend the number of digital outputs.
- void initializeOutputPins ()

4.15.1 Macro Definition Documentation

4.15.1.1 CLK_ST_PIN

```
#define CLK_ST_PIN PB0
```

Definition at line 9 of file output_pins.h.

4.15.2 Function Documentation

4.15.2.1 initializeOutputPins()

```
void initializeOutputPins ( ) [inline]
```

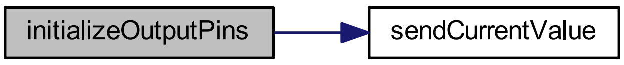
Definition at line 13 of file output_pins.h.

```
                                   {
    setBit(DDRB, CLK_ST_PIN);
    setBit(PORTB, CLK_ST_PIN);
    sendCurrentValue();
}
```

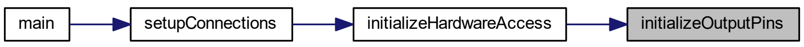
References CLK_ST_PIN, sendCurrentValue(), and setBit.

Referenced by initializeHardwareAccess().

Here is the call graph for this function:



Here is the caller graph for this function:



4.15.2.2 sendCurrentValue()

```
void sendCurrentValue ( )
```

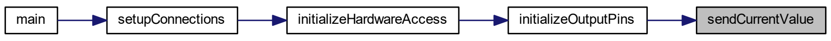
Uses a 74HC595 to extend the number of digital outputs.

Definition at line 10 of file output_pins.c.

```
                        {
    clearBit(PORTB, CLK_ST_PIN);
    sendByteOnSPI(currentValue);
    setBit(PORTB, CLK_ST_PIN);
}
```

Referenced by initializeOutputPins().

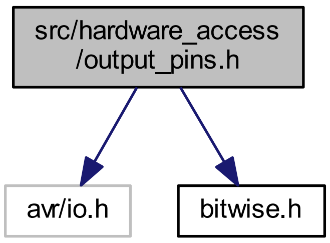
Here is the caller graph for this function:



4.16 src/hardware_access/output_pins.h File Reference

```
#include <avr/io.h>
#include "bitwise.h"
```

Include dependency graph for output_pins.h:



Macros

- #define CLK_ST_PIN PB0

Functions

- void sendCurrentValue ()

  Uses a 74HC595 to extend the number of digital outputs.
- void initializeOutputPins ()

4.16.1 Macro Definition Documentation

4.16.1.1 CLK_ST_PIN

```
#define CLK_ST_PIN PB0
```

Definition at line 9 of file output_pins.h.

4.16.2 Function Documentation

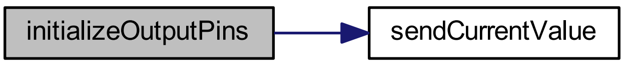
4.16.2.1 initializeOutputPins()

```
void initializeOutputPins ( )  [inline]
```

Definition at line 12 of file output_pins.h.
```
                                  {
    setBit(DDRB, CLK_ST_PIN);
    setBit(PORTB, CLK_ST_PIN);
    sendCurrentValue();
}
```

References CLK_ST_PIN, sendCurrentValue(), and setBit.

Here is the call graph for this function:



4.16.2.2 sendCurrentValue()

```
void sendCurrentValue ( )
```

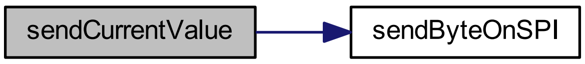
Uses a 74HC595 to extend the number of digital outputs.

Definition at line 10 of file output_pins.c.

```
10                          {
11     clearBit(PORTB, CLK_ST_PIN);
12     sendByteOnSPI(currentValue);
13     setBit(PORTB, CLK_ST_PIN);
14 }
```

References clearBit, CLK_ST_PIN, and sendByteOnSPI().

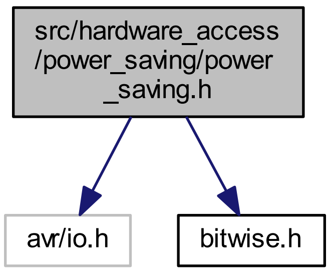
Here is the call graph for this function:



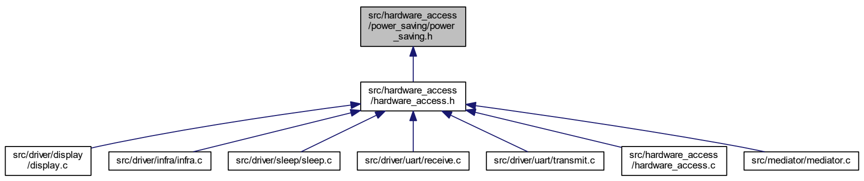
4.17 src/hardware_access/power_saving/power_saving.h File Reference

```
#include <avr/io.h>
#include "bitwise.h"
```

Include dependency graph for power_saving.h:



This graph shows which files directly or indirectly include this file:



Functions

- void initializePowerSaving ()

4.17.1 Function Documentation

4.17.1.1 initializePowerSaving()

```
void initializePowerSaving ( )  [inline]
```

disable ADC to save power

disable power to ADC (again?)

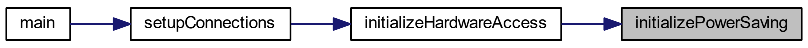
Definition at line 8 of file power_saving.h.

```
                                        {
      setBit(ACSR, ACD);    /// disable ADC to save power
       PRR = BV(PRADC);  /// disable power to ADC (again?)
 }
```

References BV, and setBit.

Referenced by initializeHardwareAccess().

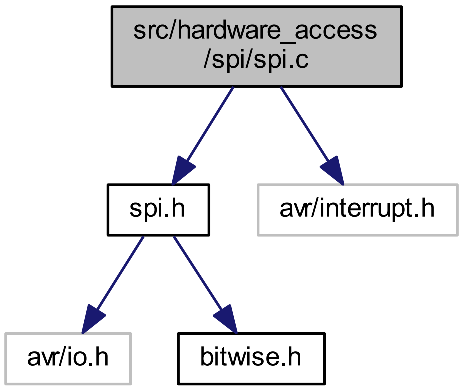
Here is the caller graph for this function:



4.18 src/hardware_access/spi/spi.c File Reference

```
#include "spi.h"
#include <avr/interrupt.h>
```

Include dependency graph for spi.c:



Functions

- void sendByteOnSPI (uint8_t byte)

4.18.1 Function Documentation

4.18.1.1 sendByteOnSPI()

```
void sendByteOnSPI (
              uint8_t byte )
```

Send a single byte on the built-in SPI interface The transfer is done in a serial manner to achieve greater throughput.

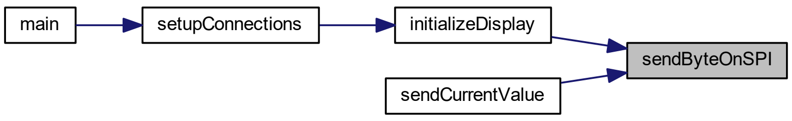
Definition at line 5 of file spi.c.

```
                                  {
    cli();
    USIDR = byte;
    for (uint8_t i = 16; i != 0; i--) {
        USICR = BV(USIWM0) | BV(USICLK) | BV(USITC) | BV(USICS1);
     }
     sei();
}
```

References BV.

Referenced by initializeDisplay(), and sendCurrentValue().

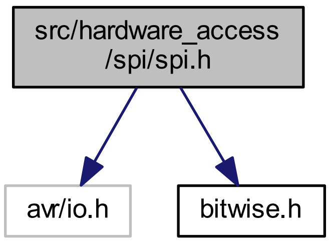
Here is the caller graph for this function:



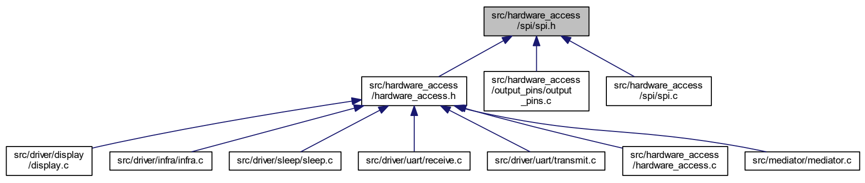
4.19 src/hardware_access/spi/spi.h File Reference

```
#include <avr/io.h>
#include "bitwise.h"
```

Include dependency graph for spi.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define DO_PIN PB1
- #define USCK_PIN PB2

Functions

- void initializeSPI ()
- void sendByteOnSPI (uint8_t byte)

4.19.1 Macro Definition Documentation

4.19.1.1 DO_PIN

```
#define DO_PIN PB1
```

Definition at line 9 of file spi.h.

4.19.1.2 USCK_PIN

```
#define USCK_PIN PB2
```

Definition at line 10 of file spi.h.

4.19.2 Function Documentation

4.19.2.1 initializeSPI()

```
void initializeSPI ( )  [inline]
```

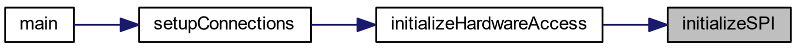
set pin directions for MOSI and SCK

Definition at line 11 of file spi.h.
```
                            {
    DDRB |= BV(DO_PIN) | BV(USCK_PIN);    /// set pin directions for MOSI and SCK
}
```

References BV, DO_PIN, and USCK_PIN.

Referenced by initializeHardwareAccess().

Here is the caller graph for this function:



4.19.2.2 sendByteOnSPI()

```
void sendByteOnSPI (
             uint8_t byte )
```

This function can be used from other sibling sub-modules it's required for the current outpu_pins implementation.

Send a single byte on the built-in SPI interface The transfer is done in a serial manner to achieve greater throughput.

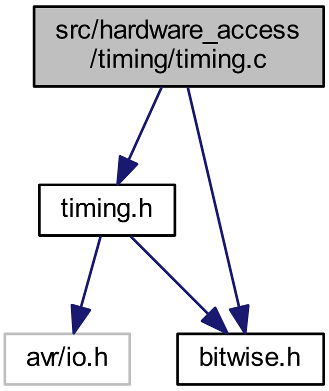
Definition at line 5 of file spi.c.
```
                                {
   cli();
   USIDR = byte;
   for (uint8_t i = 16; i != 0; i--) {
       USICR = BV(USIWM0) | BV(USICLK) | BV(USITC) | BV(USICS1);
    }
    sei();
}
```

References BV.

4.20 src/hardware_access/timing/timing.c File Reference

```
#include "timing.h"
#include "bitwise.h"
```

Include dependency graph for timing.c:



Functions

- void enableTimerA (uint8_t triggerInterruptInXTicks)

  Enable interrupt OCCRA for TIMER0 with a modulo of triggerInterruptInXTicks.
- void enableTimerB (uint8_t triggerInterruptInXTicks)

  Enable interrupt OCCRB for TIMER0B with a modulo of triggerInterruptInXTicks.
- void disableTimerB ()
- void enableFastTimerA (uint8_t triggerInterruptInXTicks)

  Enable interrupt OCCRA for TIMER1 with a modulo of triggerInterruptInXTicks.
- void disableFastTimerA ()
- uint8_t getTimestampFromFastTimer ()

  Return TCNT1.
- uint8_t getTimeSince (uint8_t timestamp)

4.20.1 Function Documentation

4.20.1.1 disableFastTimerA()

```
void disableFastTimerA ( )
```

Definition at line 25 of file timing.c.

```
25                         {
26     clearBit(TIMSK, OCIE1A);
27 }
```

References clearBit.

4.20.1.2 disableTimerB()

```
void disableTimerB ( )
```

Definition at line 15 of file timing.c.

```
            {
    clearBit(TIMSK, OCIE0B);
}
```

References clearBit.

4.20.1.3 enableFastTimerA()

```
void enableFastTimerA (
            uint8_t triggerInterruptInXTicks )
```

Enable interrupt OCCRA for TIMER1 with a modulo of triggerInterruptInXTicks.

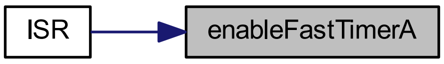
Definition at line 19 of file timing.c.

```
                                                          {
    setBit(TIFR, OCF1A);
    OCR1A = TCNT1 + triggerInterruptInXTicks;
    setBit(TIMSK, OCIE1A);
}
```

References setBit.

Referenced by ISR().

Here is the caller graph for this function:



4.20.1.4 enableTimerA()

```
void enableTimerA (
            uint8_t triggerInterruptInXTicks )
```

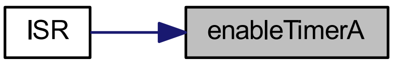
Enable interrupt OCCRA for TIMER0 with a modulo of triggerInterruptInXTicks.

Definition at line 5 of file timing.c.

```
                                             {
    OCR0A = TCNT0 + triggerInterruptInXTicks;
}
```

Referenced by ISR().

Here is the caller graph for this function:



4.20.1.5 enableTimerB()

```
void enableTimerB (
            uint8_t triggerInterruptInXTicks )
```

Enable interrupt OCCRB for TIMER0B with a modulo of triggerInterruptInXTicks.

Definition at line 9 of file timing.c.

```
                                             {
    setBit(TIFR, OCF0B);
    OCR0B = TCNT0 + triggerInterruptInXTicks;
    setBit(TIMSK, OCIE0B);
}
```

References setBit.

4.20.1.6 getTimeSince()

```
uint8_t getTimeSince (
            uint8_t timestamp )
```

Return the time since a timestamp returned by getTimestampFromFastTimer Accounts for overflow.

Definition at line 33 of file timing.c.

```
                                    {
    return timestamp <= TCNT1 ?
        TCNT1 - timestamp :
        (uint8_t)(255 - timestamp) + TCNT1;
}
```

4.20.1.7 getTimestampFromFastTimer()

```
uint8_t getTimestampFromFastTimer ( )
```

Return TCNT1.

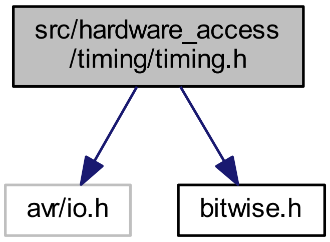
Definition at line 29 of file timing.c.

```
                              {
    return TCNT1;
}
```

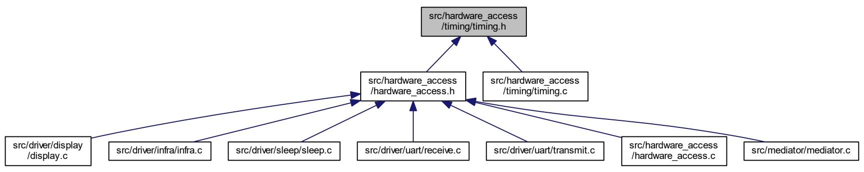
4.21 src/hardware_access/timing/timing.h File Reference

```
#include <avr/io.h>
#include "bitwise.h"
```

Include dependency graph for timing.h:



This graph shows which files directly or indirectly include this file:



Functions

- void initializeTiming ()

4.21.1 Function Documentation

4.21.1.1 initializeTiming()

```
void initializeTiming ( )  [inline]
```

CLK / 256

CLK / 128

enable compare interrupts

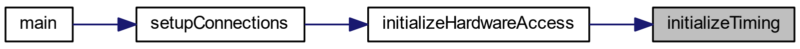
Definition at line 7 of file timing.h.

```
7                           {
8     TCCR0B = BV(CS02);                 /// CLK / 256
9     TCCR1 = BV(CS13);              /// CLK / 128
10     TIMSK = BV(OCIE0A) | BV(OCIE1A); /// enable compare interrupts
11 }
```

References BV.

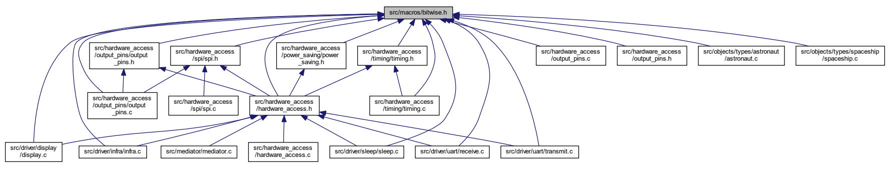
Referenced by initializeHardwareAccess().

Here is the caller graph for this function:



4.22 src/macros/bitwise.h File Reference

This graph shows which files directly or indirectly include this file:


src/macros/bitwise.h
src/hardware_access/output_pins/output_pins.h
src/hardware_access/spi/spi.h
src/hardware_access/power_saving/power_saving.h
src/hardware_access/timing/timing.h
src/hardware_access/output_pins.c
src/hardware_access/output_pins.h
src/objects/types/astronaut/astronaut.c
src/objects/types/spaceship/spaceship.c
src/hardware_access/output_pins/output_pins.c
src/hardware_access/spi/spi.c
src/hardware_access/hardware_access.h
src/hardware_access/timing/timing.c
src/driver/display/display.c
src/driver/infra/infra.c
src/mediator/mediator.c
src/hardware_access/hardware_access.c
src/driver/sleep/sleep.c
src/driver/uart/receive.c
src/driver/uart/transmit.c

Macros

- #define BV(x) (1 << (x))
- #define modifyBit(P, B, V) ((P) = ((P) & ∼BV(B)) | ((V) << B))
- #define setBit(P, B) ((P) |= BV(B))
- #define clearBit(P, B) ((P) &= ∼BV(B))
- #define toggleBit(P, B) ((P) ^= BV(B))
- #define getBit(P, B) (((P) & BV(B)) >> (B))

4.22.1 Macro Definition Documentation

4.22.1.1 BV

```
#define BV(
              x ) (1 << (x))
```

Definition at line 5 of file bitwise.h.

4.22.1.2 clearBit

```
#define clearBit(
              P,
              B ) ((P) &= ~BV(B))
```

Definition at line 8 of file bitwise.h.

4.22.1.3 getBit

```
#define getBit(
              P,
              B ) (((P) & BV(B)) >> (B))
```

Definition at line 10 of file bitwise.h.

4.22.1.4 modifyBit

```
#define modifyBit(
              P,
              B,
              V ) ((P) = ((P) & ~BV(B)) | ((V) << B))
```

Definition at line 6 of file bitwise.h.

4.22.1.5 setBit

```
#define setBit(
              P,
              B ) ((P) |= BV(B))
```

Definition at line 7 of file bitwise.h.

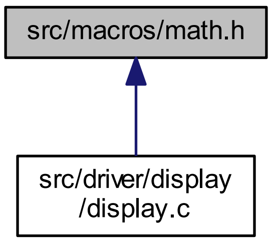
4.22.1.6 toggleBit

```
#define toggleBit(
              P,
              B ) ((P) ^= BV(B))
```

Definition at line 9 of file bitwise.h.

4.23 src/macros/math.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define max(a, b) ((a) > (b) ? (a) : (b))
- #define min(a, b) ((a) > (b) ? (b) : (a))
- #define abs(a, b) ((a) > 0 ? (a) : (-a))

4.23.1 Macro Definition Documentation

4.23.1.1 abs

```
#define abs(
              a,
              b ) ((a) > 0 ? (a) : (-a))
```

Definition at line 8 of file math.h.

4.23.1.2 max

```
#define max(
            a,
            b ) ((a) > (b) ? (a) : (b))
```

Definition at line 6 of file math.h.

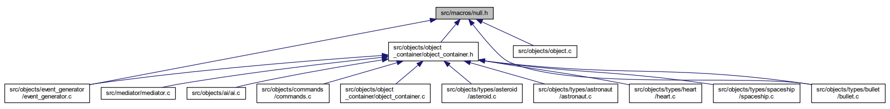
4.23.1.3 min

```
#define min(
            a,
            b ) ((a) > (b) ? (b) : (a))
```

Definition at line 7 of file math.h.

4.24 src/macros/null.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define NULL ((void∗)0)

4.24.1 Macro Definition Documentation

4.24.1.1 NULL

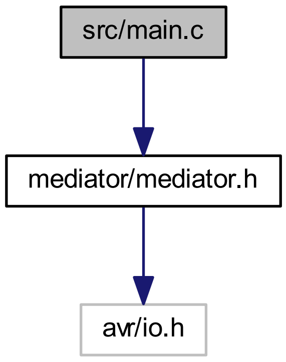
```
#define NULL ((void*)0)
```

Definition at line 3 of file null.h.

4.25 src/main.c File Reference

```
#include "mediator/mediator.h"
```

Include dependency graph for main.c:



Functions

- int main (void)

4.25.1 Function Documentation

4.25.1.1 main()

```
int main (
            void )
```

Stemming from the module oriented nature of the project there is a module responsible for setting up and orchestrating the other modules.

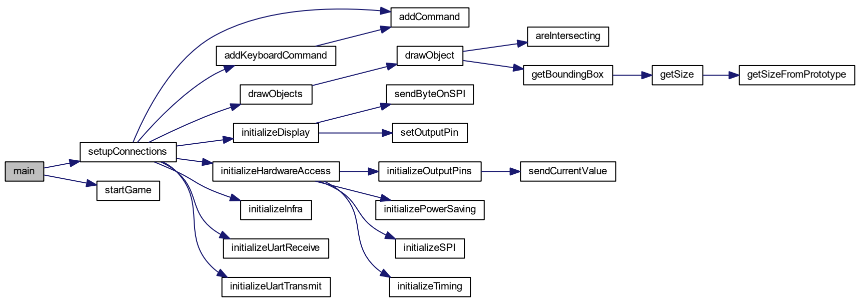
From the main function we only have to instruct the aforementioned mediator module to do its job.

Definition at line 11 of file main.c.

```
            {
    setupConnections();
    startGame();
}
```

References setupConnections(), and startGame().

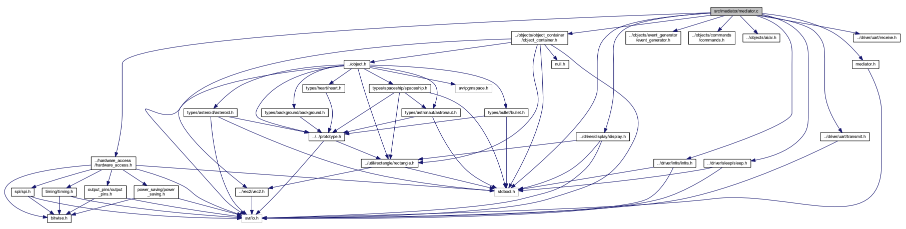
Here is the call graph for this function:



4.26 src/mediator/mediator.c File Reference

```
#include "mediator.h"
#include <stdbool.h>
#include "../hardware_access/hardware_access.h"
#include "../objects/object_container/object_container.h"
#include "../objects/event_generator/event_generator.h"
#include "../objects/commands/commands.h"
#include "../objects/ai/ai.h"
#include "../driver/display/display.h"
#include "../driver/infra/infra.h"
#include "../driver/sleep/sleep.h"
#include "../driver/uart/transmit.h"
#include "../driver/uart/receive.h"
```

Include dependency graph for mediator.c:



Macros

- #define TARGET_FRAME_DURATION 20
- #define DEATH_SCREEN_LENGTH 50
- #define REPORT_INTERVAL 50

Functions

- void addKeyboardCommand (uint8_t key)
- void setupConnections ()
- void startGame ()

  Start drawing frames and ticking objects.
- void changeDisplayContrast (int8_t by)

4.26.1 Macro Definition Documentation

4.26.1.1 DEATH_SCREEN_LENGTH

```
#define DEATH_SCREEN_LENGTH 50
```

Definition at line 19 of file mediator.c.

4.26.1.2 REPORT_INTERVAL

```
#define REPORT_INTERVAL 50
```

Definition at line 20 of file mediator.c.

4.26.1.3 TARGET_FRAME_DURATION

```
#define TARGET_FRAME_DURATION 20
```

Definition at line 18 of file mediator.c.

4.26.2 Function Documentation

4.26.2.1 addKeyboardCommand()

```
void addKeyboardCommand (
            uint8_t key )
```

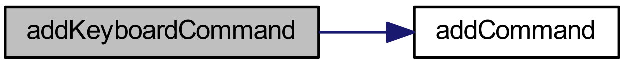
Definition at line 67 of file mediator.c.

```
                                     {
    switch (key) {
        case 'w':
            addCommand(moveUp);
            break;
        case 'a':
            addCommand(moveLeft);
            break;
        case 's':
            addCommand(moveDown);
            break;
        case 'd':
            addCommand(moveRight);
            break;
        case ' ':
            addCommand(action);
            break;
        case '+':
            addCommand(increaseContrast);
            break;
        case '-':
            addCommand(decreaseContrast);
            break;
    }
}
```

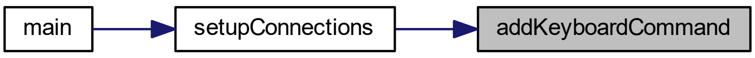
References action, addCommand(), decreaseContrast, increaseContrast, moveDown, moveLeft, moveRight, and moveUp.

Referenced by setupConnections().

Here is the call graph for this function:



Here is the caller graph for this function:



4.26.2.2 changeDisplayContrast()

```
void changeDisplayContrast (
             int8_t by )
```

Increase or decrease the contrast (brightness) of the display by the given value The contrast can be any number between 0 and 255. The function automatically clamps the contrast.

Definition at line 108 of file mediator.c.

```
108                                         {
109     if (by < 0) {
110         state.contrast = (state.contrast < -by) ? 0 : (state.contrast + by);
111     } else {
112         state.contrast = (state.contrast > 255 - by) ? 255 : (state.contrast + by);
113     }
114 
115     setDisplayContrast(state.contrast);
116 }
```

4.26.2.3 setupConnections()

```
void setupConnections ( )
```

Setup the drivers, and business layer objects and their relations It is kind of a very basic dependency injection.

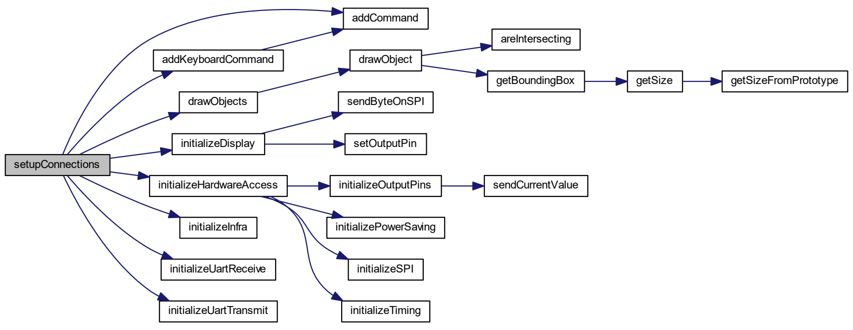
Definition at line 93 of file mediator.c.

```
93                       {
94     initializeHardwareAccess();
95     initializeInfra(addCommand);
96     initializeDisplay(drawObjects);
97     initializeUartTransmit();
98     initializeUartReceive(addKeyboardCommand);
99 }
```

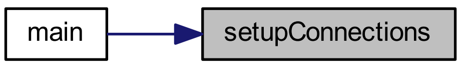
References addCommand(), addKeyboardCommand(), drawObjects(), initializeDisplay(), initializeHardware←
Access(), initializeInfra(), initializeUartReceive(), and initializeUartTransmit().

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



4.26.2.4 startGame()

```
void startGame ( )
```

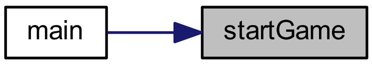
Start drawing frames and ticking objects.

Definition at line 101 of file mediator.c.

```
                {
    while (true) {
        startGameLogic();
        startFrameLoop(frameFunction, TARGET_FRAME_DURATION);
    }
}
```

Referenced by main().

Here is the caller graph for this function:



4.26.3 Variable Documentation

4.26.3.1 contrast

```
uint8_t contrast
```

Definition at line 22 of file mediator.c.

4.26.3.2 deathDownCounter

```
uint8_t deathDownCounter
```

Definition at line 25 of file mediator.c.

4.26.3.3 framesSinceLastReport

```
uint8_t framesSinceLastReport
```

Definition at line 23 of file mediator.c.

4.26.3.4 longestFrameTimeSinceLastReport

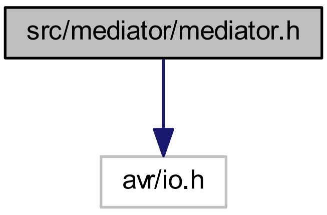
```
uint8_t longestFrameTimeSinceLastReport
```

Definition at line 24 of file mediator.c.

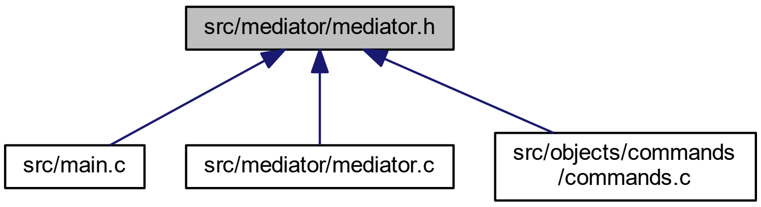
4.27 src/mediator/mediator.h File Reference

```
#include <avr/io.h>
```

Include dependency graph for mediator.h:



This graph shows which files directly or indirectly include this file:



Functions

- void setupConnections ()
- void startGame ()

  Start drawing frames and ticking objects.
- void changeDisplayContrast (int8_t by)

4.27.1 Function Documentation

4.27.1.1 changeDisplayContrast()

```
void changeDisplayContrast (
             int8_t by )
```

Increase or decrease the contrast (brightness) of the display by the given value The contrast can be any number between 0 and 255. The function automatically clamps the contrast.

Definition at line 108 of file mediator.c.

```
108                                      {
109     if (by < 0) {
110         state.contrast = (state.contrast < -by) ? 0 : (state.contrast + by);
111     } else {
112         state.contrast = (state.contrast > 255 - by) ? 255 : (state.contrast + by);
113     }
114 
115     setDisplayContrast(state.contrast);
116 }
```

4.27.1.2 setupConnections()

```
void setupConnections ( )
```

Setup the drivers, and business layer objects and their relations It is kind of a very basic dependency injection.

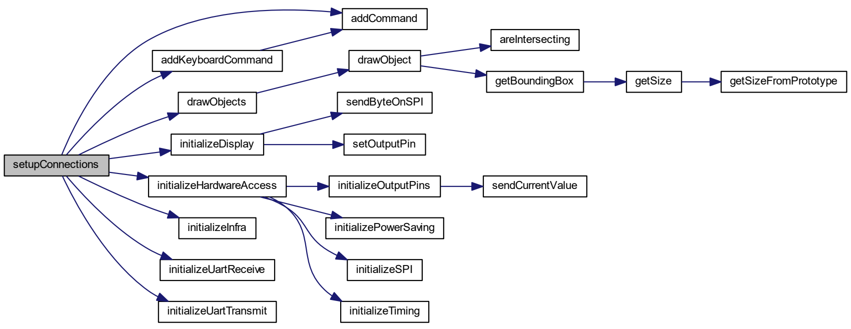
Definition at line 93 of file mediator.c.

```
93                         {
94     initializeHardwareAccess();
95     initializeInfra(addCommand);
96     initializeDisplay(drawObjects);
97     initializeUartTransmit();
98     initializeUartReceive(addKeyboardCommand);
99 }
```

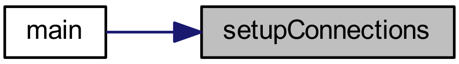
References addCommand(), addKeyboardCommand(), drawObjects(), initializeDisplay(), initializeHardware←Access(), initializeInfra(), initializeUartReceive(), and initializeUartTransmit().

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



4.27.1.3 startGame()

```
void startGame ( )
```

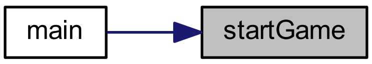
Start drawing frames and ticking objects.

Definition at line 101 of file mediator.c.

```
{
    while (true) {
        startGameLogic();
        startFrameLoop(frameFunction, TARGET_FRAME_DURATION);
    }
}
```

Referenced by main().

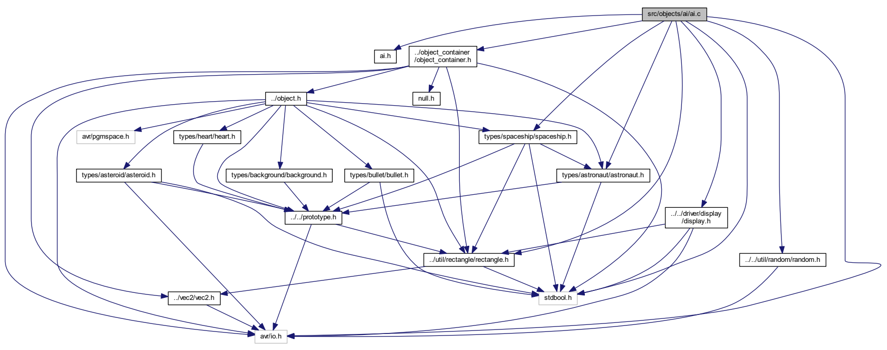
Here is the caller graph for this function:



4.28 src/objects/ai/ai.c File Reference

```
#include "ai.h"
#include <stdbool.h>
#include <avr/io.h>
#include "../object_container/object_container.h"
#include "../types/astronaut/astronaut.h"
#include "../types/spaceship/spaceship.h"
#include "../../util/rectangle/rectangle.h"
#include "../../util/random/random.h"
```

```
#include "../../driver/display/display.h"
```
Include dependency graph for ai.c:



Data Structures

- struct AIAction

Macros

- #define AI_ACTION_COUNT 5

Typedefs

- typedef bool(∗ Predicate) (Rectangle ∗, Object ∗)
- typedef void(∗ Execution) (Object ∗)

Functions

- void handleAI ()

4.28.1 Macro Definition Documentation

4.28.1.1 AI_ACTION_COUNT

```
#define AI_ACTION_COUNT 5
```

Definition at line 15 of file ai.c.

4.28.2 Typedef Documentation

4.28.2.1 Execution

```
typedef void(* Execution) (Object *)
```

Definition at line 17 of file ai.c.

4.28.2.2 Predicate

```
typedef bool(* Predicate) (Rectangle *, Object *)
```

Definition at line 16 of file ai.c.

4.28.3 Function Documentation

4.28.3.1 handleAI()

```
void handleAI ( )
```

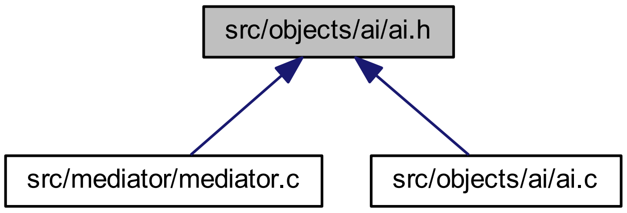
If there are non player controlled astronauts control them according to some basic rule set

Definition at line 202 of file ai.c.

```
                {
    timeSinceLastAction++;

    for (uint8_t j = 0; j < ACTION_COUNT; j++) {
        actions[j].isSomeoneDoingThis = false;
    }

    for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
        if (objects[i].prototype == &Astronaut && objects + i != character) {
            for (uint8_t j = 0; j < ACTION_COUNT; j++) {
                AIAction* currentAction = actions + j;
                Rectangle boundingBox = getBoundingBoxOfSpaceshipPart(currentAction->spaceshipPart);
                Object* astronautIntersectingBoundingBox = getIntersectingObjectOfType(boundingBox,
       &Astronaut);
                if (
                    isSpaceshipPartActivated(currentAction->spaceshipPart) &&
                    (!currentAction->onlyOneAstronautCanDoIt || (!currentAction->isSomeoneDoingThis &&
       astronautIntersectingBoundingBox != character)) &&
                     currentAction->predicate(&boundingBox, objects + i)
                ) {
                    if (!areIntersecting(boundingBox, getBoundingBox(objects + i))) {
                        carefullyMoveAstronaut(objects + i, add(getCenter(boundingBox),
       currentAction->deltaCenter));
                    } else if (timeSinceLastAction > AI_ACTION_INTERVAL) {
                        currentAction->execution(objects + i);
                    }
                    currentAction->isSomeoneDoingThis = true;
                    break;
                }
            }
        }
    }
}
```

4.29 src/objects/ai/ai.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define AI_ACTION_INTERVAL 15

Functions

- void handleAI ()

4.29.1 Macro Definition Documentation

4.29.1.1 AI_ACTION_INTERVAL

```
#define AI_ACTION_INTERVAL 15
```

Between AI astronauts do actions there has to be at least this many frames

Definition at line 7 of file ai.h.

4.29.2 Function Documentation

4.29.2.1 handleAI()

```
void handleAI ( )
```

If there are non player controlled astronauts control them according to some basic rule set

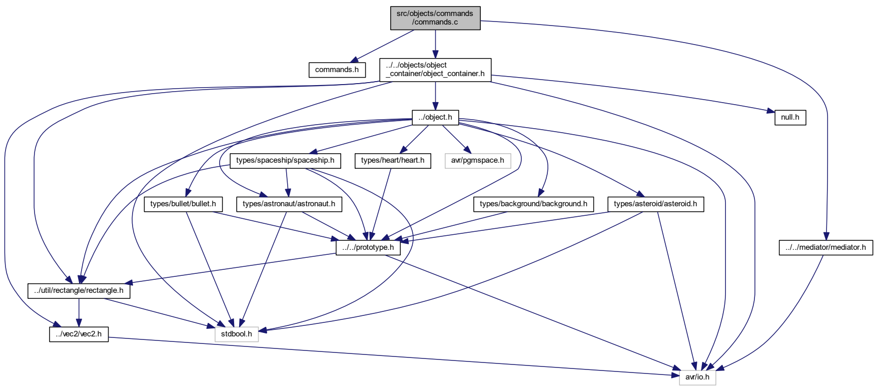
Definition at line 202 of file ai.c.

```
202                 {
203     timeSinceLastAction++;
204 
205     for (uint8_t j = 0; j < ACTION_COUNT; j++) {
206         actions[j].isSomeoneDoingThis = false;
207     }
208 
209     for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
210         if (objects[i].prototype == &Astronaut && objects + i != character) {
211             for (uint8_t j = 0; j < ACTION_COUNT; j++) {
212                 AIAction* currentAction = actions + j;
213                 Rectangle boundingBox = getBoundingBoxOfSpaceshipPart(currentAction->spaceshipPart);
214                 Object* astronautIntersectingBoundingBox = getIntersectingObjectOfType(boundingBox,
       &Astronaut);
215                 if (
216                     isSpaceshipPartActivated(currentAction->spaceshipPart) &&
217                     (!currentAction->onlyOneAstronautCanDoIt || (!currentAction->isSomeoneDoingThis &&
       astronautIntersectingBoundingBox != character)) &&
218                      currentAction->predicate(&boundingBox, objects + i)
219                 ) {
220                     if (!areIntersecting(boundingBox, getBoundingBox(objects + i))) {
221                         carefullyMoveAstronaut(objects + i, add(getCenter(boundingBox),
       currentAction->deltaCenter));
222                     } else if (timeSinceLastAction > AI_ACTION_INTERVAL) {
223                         currentAction->execution(objects + i);
224                     }
225                     currentAction->isSomeoneDoingThis = true;
226                     break;
227                 }
228             }
229         }
230     }
231 }
```

4.30 src/objects/commands/commands.c File Reference

```
#include "commands.h"
#include "../../objects/object_container/object_container.h"
#include "../../mediator/mediator.h"
```

Include dependency graph for commands.c:


src/objects/commands
/commands.c
commands.h
../../objects/object
_container/object_container.h
../object.h
null.h
types/spaceship/spaceship.h
types/heart/heart.h
avr/pgmspace.h
types/bullet/bullet.h
types/astronaut/astronaut.h
types/background/background.h
types/asteroid/asteroid.h
../../prototype.h
../../mediator/mediator.h
../util/rectangle/rectangle.h
../vec2/vec2.h
stdbool.h
avr/io.h

Functions

- void addCommand (Command command)
- void handleCommands ()

  Process every command in the buffer at once in a FIFO manner.

4.30.1 Function Documentation

4.30.1.1 addCommand()

```
void addCommand (
            Command command )
```

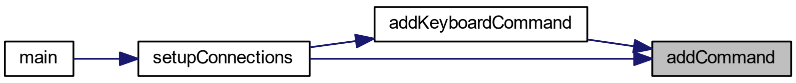
Add a new command to the buffer It will not be processed immediately.

Definition at line 25 of file commands.c.

```
                                  {
    commands.received[commands.end++] = command;
    commands.end %= COMMAND_BUFFER_SIZE;
}
```

Referenced by addKeyboardCommand(), and setupConnections().

Here is the caller graph for this function:



4.30.1.2 handleCommands()

```
void handleCommands ( )
```

Process every command in the buffer at once in a FIFO manner.

Definition at line 30 of file commands.c.

```
                       {
    while(areThereAnyCommandsLeft()) {
        Command next = getNextCommand();
        if (next == repeat) {
            next = commands.previous;
        } else {
            commands.previous = next;
        }

        switch(next) {
            case increaseContrast:
                changeDisplayContrast(CONTRAST_STEP);
                break;
            case decreaseContrast:
                changeDisplayContrast(-CONTRAST_STEP);
                break;
            case moveLeft:
                moveAstronaut(character, directions[west]);
                break;
            case moveRight:
                moveAstronaut(character, directions[east]);
                break;
            case moveUp:
                moveAstronaut(character, directions[north]);
                break;
            case moveDown:
                moveAstronaut(character, directions[south]);
                break;
            case action:
                makeAstronautDoAction(character);
                commands.previous = noAction;
                break;
            default:
                break;
        }
    }
}
```

4.30.2 Variable Documentation

4.30.2.1 end

```
uint8_t end
```

Definition at line 10 of file commands.c.

4.30.2.2 previous

```
Command previous
```

Definition at line 11 of file commands.c.

4.30.2.3 received

```
Command received[COMMAND_BUFFER_SIZE]
```

Definition at line 8 of file commands.c.

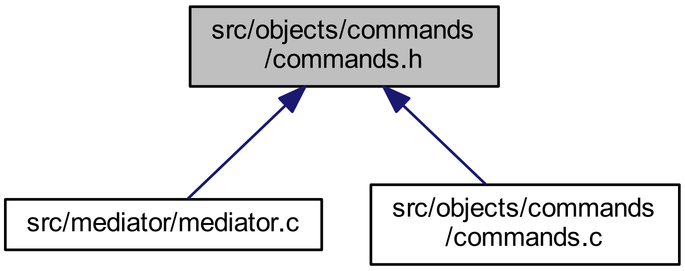
4.30.2.4 start

```
uint8_t start
```

Definition at line 9 of file commands.c.

4.31 src/objects/commands/commands.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define COMMAND_BUFFER_SIZE 8
- #define CONTRAST_STEP 15

Enumerations

- enum Command {
  noCommand = 0, repeat = 1, increaseContrast = 87, decreaseContrast = 31,
  moveUp = 231, moveDown = 181, moveLeft = 239, moveRight = 165,
  action = 199 }

Functions

- void addCommand (Command command)
- void handleCommands ()

  Process every command in the buffer at once in a FIFO manner.

4.31.1 Macro Definition Documentation

4.31.1.1 COMMAND_BUFFER_SIZE

```
#define COMMAND_BUFFER_SIZE 8
```

There can be no more than COMMAND_BUFFER_SIZE commands waiting for processing simultaneously

Definition at line 7 of file commands.h.

4.31.1.2 CONTRAST_STEP

```
#define CONTRAST_STEP 15
```

increaseContrast and decreaseContrast changes the contrast with this value

Definition at line 11 of file commands.h.

4.31.2 Enumeration Type Documentation

4.31.2.1 Command

```
enum Command
```

The possible inputs of the system Coincidentally these are the codes of the IR remote controller's buttons.

Enumerator

noCommand	
repeat	
increaseContrast	
decreaseContrast	
moveUp	
moveDown	
moveLeft	
moveRight	
action	

Definition at line 15 of file commands.h.

```
               {
    noCommand = 0,
    repeat = 1,
    increaseContrast = 87,
    decreaseContrast = 31,
    moveUp = 231,
    moveDown = 181,
    moveLeft = 239,
    moveRight = 165,
    action = 199,
} Command;
```

4.31.3 Function Documentation

4.31.3.1 addCommand()

```
void addCommand (
               Command command )
```

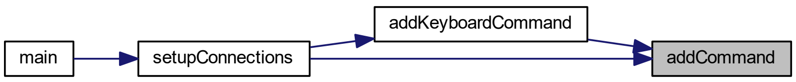
Add a new command to the buffer It will not be processed immediately.

Definition at line 25 of file commands.c.

```
                                    {
    commands.received[commands.end++] = command;
    commands.end %= COMMAND_BUFFER_SIZE;
}
```

Referenced by addKeyboardCommand(), and setupConnections().

Here is the caller graph for this function:



4.31.3.2 handleCommands()

```
void handleCommands ( )
```

Process every command in the buffer at once in a FIFO manner.

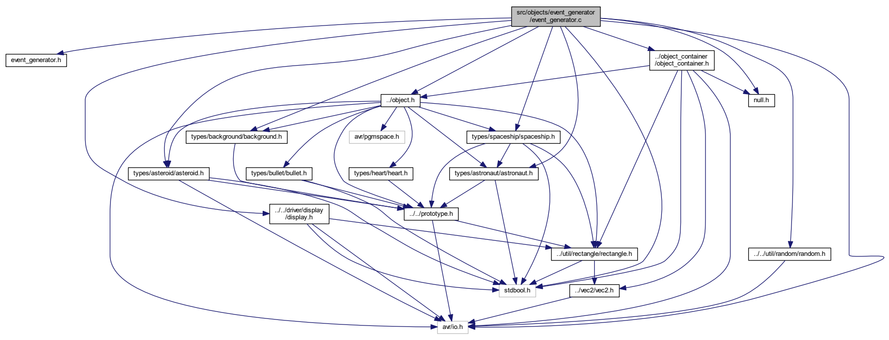
Definition at line 30 of file commands.c.

```
                          {
    while(areThereAnyCommandsLeft()) {
        Command next = getNextCommand();
        if (next == repeat) {
            next = commands.previous;
        } else {
            commands.previous = next;
        }

        switch(next) {
            case increaseContrast:
                changeDisplayContrast(CONTRAST_STEP);
                break;
            case decreaseContrast:
                changeDisplayContrast(-CONTRAST_STEP);
                break;
            case moveLeft:
                moveAstronaut(character, directions[west]);
                break;
            case moveRight:
                moveAstronaut(character, directions[east]);
                break;
            case moveUp:
                moveAstronaut(character, directions[north]);
                break;
            case moveDown:
                moveAstronaut(character, directions[south]);
                break;
            case action:
                makeAstronautDoAction(character);
                commands.previous = noAction;
                break;
            default:
                break;
        }
    }
}
```

4.32 src/objects/event_generator/event_generator.c File Reference

```
#include "event_generator.h"
#include <avr/io.h>
#include <stdbool.h>
#include "../object.h"
#include "../object_container/object_container.h"
#include "../types/spaceship/spaceship.h"
#include "../types/background/background.h"
#include "../types/astronaut/astronaut.h"
#include "../types/asteroid/asteroid.h"
#include "null.h"
#include "../../util/random/random.h"
#include "../../driver/display/display.h"
```

Include dependency graph for event_generator.c:


src/objects/event_generator/event_generator.c
event_generator.h
../object_container/object_container.h
null.h
../object.h
types/background/background.h
avr/pgmspace.h
types/spaceship/spaceship.h
types/asteroid/asteroid.h
types/bullet/bullet.h
types/heart/heart.h
types/astronaut/astronaut.h
../../driver/display/display.h
../../prototype.h
../util/rectangle/rectangle.h
../../util/random/random.h
stdbool.h
../vec2/vec2.h
avr/io.h

Typedefs

- typedef bool(∗ Predicate) (Rectangle ∗)

Functions

- bool generateAstronautPredicate (Rectangle ∗proposedBoundingBox)
- bool generateAsteroidPredicate (Rectangle ∗proposedBoundingBox)
- void createObjects ()
- void generateEvents ()

    Generate asteroids and astronaut randomly based on a set of conditions.

4.32.1 Typedef Documentation

4.32.1.1 Predicate

```
typedef bool(∗ Predicate) (Rectangle ∗)
```

Definition at line 17 of file event_generator.c.

4.32.2 Function Documentation

4.32.2.1 createObjects()

```
void createObjects ( )
```

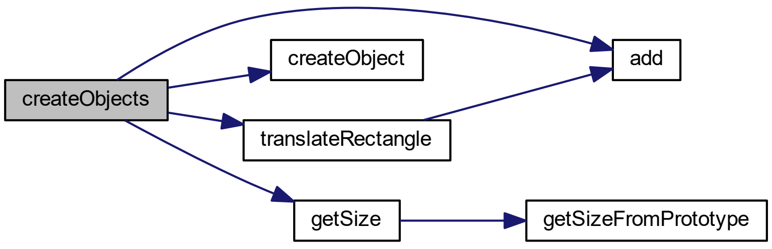
Create the necessary objects in order to start the game These include the background, spaceship and the player's character.

Definition at line 64 of file event_generator.c.

```
                         {
    createObject(&Background, getEmptyObjectSpace());

    createObject(&Spaceship, spaceshipObject);
    spaceshipObject->position = (Vec2){EXHAUST_BOUNDING_BOX.size.x, DISPLAY_HEIGHT_IN_PIXELS / 2 -
    getSize(spaceshipObject).y / 2};

    createObject(&Astronaut, character);
    Rectangle upperFloor = translateRectangle(UPPER_FLOOR_BOUNDING_BOX, spaceshipObject->position);
    character->position = add(upperFloor.position, (Vec2){10, 1});
}
```

References add(), Astronaut, Background, character, createObject(), DISPLAY_HEIGHT_IN_PIXELS, EXH←
AUST_BOUNDING_BOX, getEmptyObjectSpace, getSize(), Rectangle::position, Spaceship, spaceshipObject, translateRectangle(), UPPER_FLOOR_BOUNDING_BOX, and Vec2::y.

Here is the call graph for this function:



4.32.2.2 generateAsteroidPredicate()

```
bool generateAsteroidPredicate (
             Rectangle * proposedBoundingBox )
```

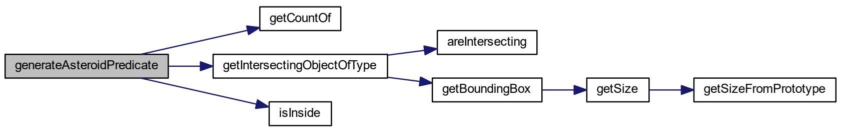
Definition at line 55 of file event_generator.c.

```
                                                                    {
    return (
        getCountOf(&Asteroid) < MAX_ASTEROID_COUNT &&
        isInside(*proposedBoundingBox, WINDOW) &&
        getIntersectingObjectOfType(*proposedBoundingBox, &Spaceship) == NULL &&
        getIntersectingObjectOfType(*proposedBoundingBox, &Asteroid) == NULL
    );
```

```
62 }
```

References Asteroid, getCountOf(), getIntersectingObjectOfType(), isInside(), MAX_ASTEROID_COUNT, NULL, Spaceship, and WINDOW.

Here is the call graph for this function:



4.32.2.3 generateAstronautPredicate()

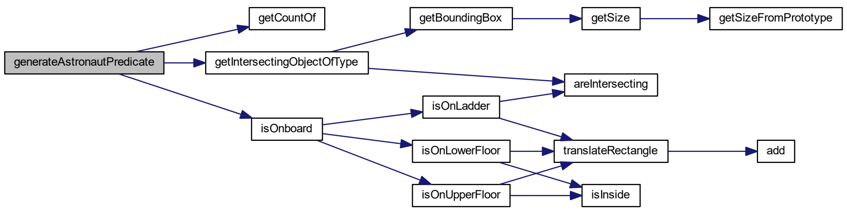
```
bool generateAstronautPredicate (
            Rectangle * proposedBoundingBox )
```

Definition at line 44 of file event_generator.c.

```
44                                                                {
45      return (
46          (
47              (getCountOf(&Astronaut) == 1 && spaceshipObject->as.spaceship.progress >= hasTable) ||
48              (getCountOf(&Astronaut) == 2 && spaceshipObject->as.spaceship.progress >= hasFullCrew)
49          ) &&
50          getIntersectingObjectOfType(*proposedBoundingBox, &Astronaut) == NULL &&
51          isOnboard(*proposedBoundingBox)
52      );
53 }
```

References Astronaut, getCountOf(), getIntersectingObjectOfType(), hasFullCrew, hasTable, isOnboard(), NULL, and spaceshipObject.

Here is the call graph for this function:



4.32.2.4 generateEvents()

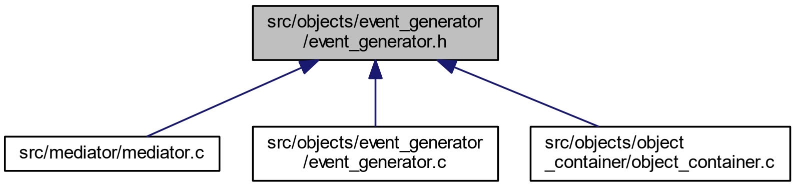
```
void generateEvents ( )
```

Generate asteroids and astronaut randomly based on a set of conditions.

Definition at line 75 of file event_generator.c.
```
                         {
       generate(&Astronaut, generateAstronautPredicate);
       generate(&Asteroid, generateAsteroidPredicate);
   }
```

4.33 src/objects/event_generator/event_generator.h File Reference

This graph shows which files directly or indirectly include this file:


src/objects/event_generator
/event_generator.h
src/mediator/mediator.c
src/objects/event_generator
/event_generator.c
src/objects/object
_container/object_container.c

Macros

- #define MAX_ASTEROID_COUNT 2
- #define TRY_COUNT 16

Functions

- void createObjects ()
- void generateEvents ()

  Generate asteroids and astronaut randomly based on a set of conditions.

4.33.1 Macro Definition Documentation

4.33.1.1 MAX_ASTEROID_COUNT

```
#define MAX_ASTEROID_COUNT 2
```

Definition at line 5 of file event_generator.h.

4.33.1.2 TRY_COUNT

```
#define TRY_COUNT 16
```

For minimizing code size the position of generated objects is decided randomly. If it fits then it stays. For each generated object can be a maximum of TRY_COUNT tries.

Definition at line 10 of file event_generator.h.

4.33.2 Function Documentation

4.33.2.1 createObjects()

```
void createObjects ( )
```

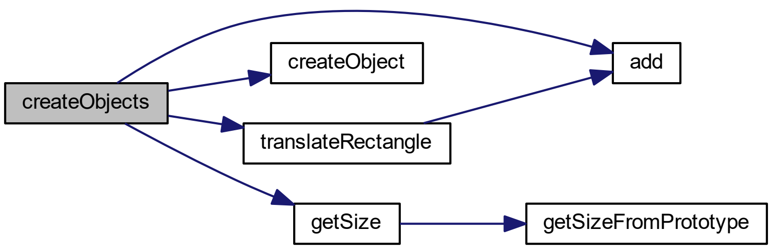
Create the necessary objects in order to start the game These include the background, spaceship and the player's character.

Definition at line 64 of file event_generator.c.

```
                       {
    createObject(&Background, getEmptyObjectSpace());

    createObject(&Spaceship, spaceshipObject);
    spaceshipObject->position = (Vec2){EXHAUST_BOUNDING_BOX.size.x, DISPLAY_HEIGHT_IN_PIXELS / 2 -
    getSize(spaceshipObject).y / 2};

    createObject(&Astronaut, character);
    Rectangle upperFloor = translateRectangle(UPPER_FLOOR_BOUNDING_BOX, spaceshipObject->position);
    character->position = add(upperFloor.position, (Vec2){10, 1});
}
```

References add(), Astronaut, Background, character, createObject(), DISPLAY_HEIGHT_IN_PIXELS, EXH↩ AUST_BOUNDING_BOX, getEmptyObjectSpace, getSize(), Rectangle::position, Spaceship, spaceshipObject, translateRectangle(), UPPER_FLOOR_BOUNDING_BOX, and Vec2::y.

Here is the call graph for this function:



4.33.2.2 generateEvents()

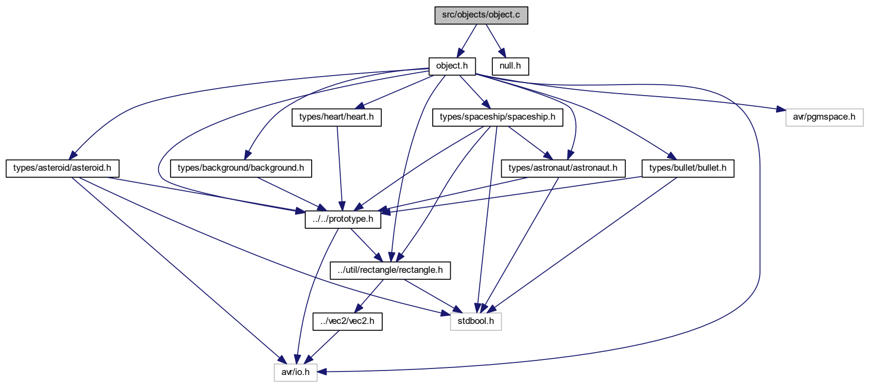
```
void generateEvents ( )
```

Generate asteroids and astronaut randomly based on a set of conditions.

Definition at line 75 of file event_generator.c.
```
         {
    generate(&Astronaut, generateAstronautPredicate);
    generate(&Asteroid, generateAsteroidPredicate);
}
```

4.34 src/objects/object.c File Reference

```
#include "object.h"
#include "null.h"
```
Include dependency graph for object.c:


src/objects/object.c
object.h
null.h
types/heart/heart.h
types/spaceship/spaceship.h
avr/pgmspace.h
types/asteroid/asteroid.h
types/background/background.h
types/astronaut/astronaut.h
types/bullet/bullet.h
../../prototype.h
../util/rectangle/rectangle.h
../vec2/vec2.h
stdbool.h
avr/io.h

Functions

- Object ∗ createObject (Prototype const ∗prototype, Object ∗holder)
- void tickObject (Object ∗object, uint8_t previousFrameTime)
- void drawObject (Object ∗object, Rectangle compositingWindow)
- Vec2 getSizeFromPrototype (Prototype const ∗prototype)
- Vec2 getSize (Object const ∗object)

  Find out the prototype of the object and return the size of that.
- void move (Object ∗object, Vec2 value)

  Move the position of the object by a vector.
- Rectangle getBoundingBox (Object const ∗object)

4.34.1 Function Documentation

4.34.1.1 createObject()

```
Object* createObject (
            Prototype const * prototype,
            Object * holder )
```

Set the prototype of the holder and initialize all the holder's vale to zero. Return the freshly updated holder

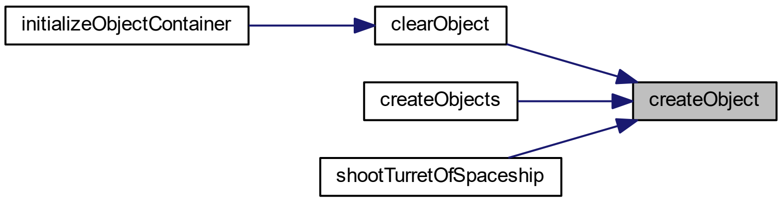
Definition at line 5 of file object.c.

```
                                                       {
    Object empty = {0};
    *holder = empty;
    holder->prototype = prototype;
    return holder;
}
```

References _object_t::prototype.

Referenced by clearObject(), createObjects(), and shootTurretOfSpaceship().

Here is the caller graph for this function:



4.34.1.2 drawObject()

```
void drawObject (
            Object * object,
            Rectangle compositingWindow )
```

Call the draw function referenced in the object's prototype on the object itself. Does nothing when called with NULL.

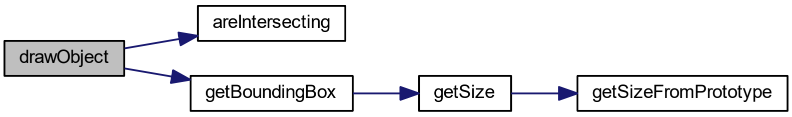
Definition at line 18 of file object.c.

```
                                                     {
    if (object->prototype != NULL && areIntersecting(getBoundingBox(object), compositingWindow)) {
        ((DrawMethod)pgm_read_word(&object->prototype->draw))(object, compositingWindow);
    }
}
```

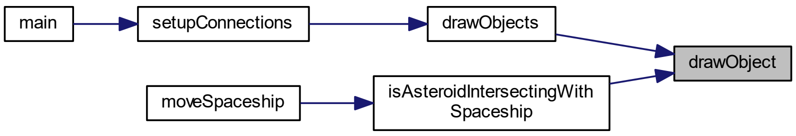
References areIntersecting(), compositingWindow, Prototype::draw, getBoundingBox(), NULL, and _object_t↩
::prototype.

Referenced by drawObjects(), and isAsteroidIntersectingWithSpaceship().

Here is the call graph for this function:



Here is the caller graph for this function:



4.34.1.3 getBoundingBox()

```
Rectangle getBoundingBox (
            Object const * object )
```

Get a new rectangle from the objects position and its prototype's size

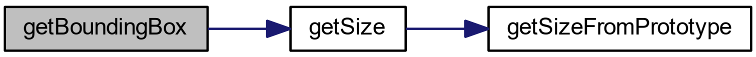
Definition at line 39 of file object.c.

```
                                          {
    return (Rectangle){object->position, getSize(object)};
}
```

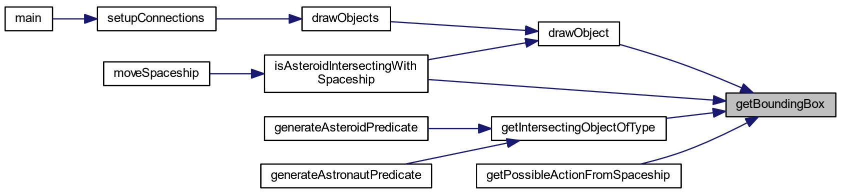
References getSize(), and Rectangle::position.

Referenced by drawObject(), getIntersectingObjectOfType(), getPossibleActionFromSpaceship(), and isAsteroid↵ IntersectingWithSpaceship().

Here is the call graph for this function:



Here is the caller graph for this function:


main
setupConnections
drawObjects
drawObject
moveSpaceship
isAsteroidIntersectingWith Spaceship
getBoundingBox
generateAsteroidPredicate
getIntersectingObjectOfType
generateAstronautPredicate
getPossibleActionFromSpaceship

4.34.1.4 getSize()

```
Vec2 getSize (
            Object const * object )
```

Find out the prototype of the object and return the size of that.

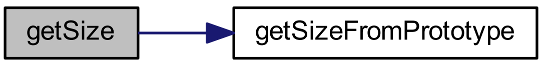
Definition at line 31 of file object.c.

```
31                          {
32     return getSizeFromPrototype(object->prototype);
33 }
```

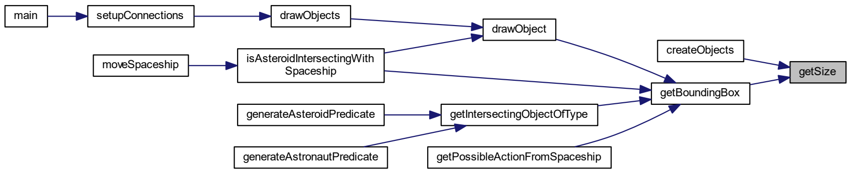
References getSizeFromPrototype(), and _object_t::prototype.

Referenced by createObjects(), and getBoundingBox().

Here is the call graph for this function:



Here is the caller graph for this function:


main
setupConnections
drawObjects
drawObject
moveSpaceship
isAsteroidIntersectingWith Spaceship
createObjects
getSize
getBoundingBox
generateAsteroidPredicate
getIntersectingObjectOfType
generateAstronautPredicate
getPossibleActionFromSpaceship

4.34.1.5 getSizeFromPrototype()

```
Vec2 getSizeFromPrototype (
            Prototype const * prototype )
```

required for casting

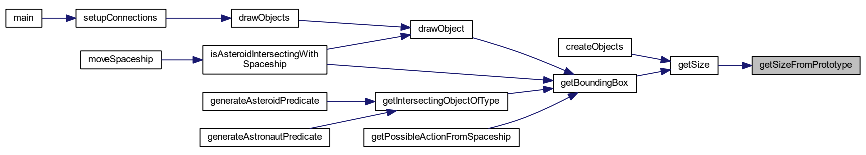
Definition at line 24 of file object.c.

```
24                                                    {
25     /// required for casting
26     uint16_t read = pgm_read_word(&prototype->size);
27     Vec2* v = (Vec2*) &read;
28     return *v;
29 }
```

References Prototype::size.

Referenced by getSize().

Here is the caller graph for this function:


main
setupConnections
drawObjects
drawObject
moveSpaceship
isAsteroidIntersectingWith Spaceship
createObjects
getSize
getSizeFromPrototype
getBoundingBox
generateAsteroidPredicate
getIntersectingObjectOfType
generateAstronautPredicate
getPossibleActionFromSpaceship

4.34.1.6 move()

```
void move (
            Object * object,
            Vec2 value )
```

Move the position of the object by a vector.

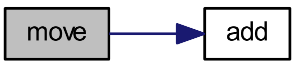
Definition at line 35 of file object.c.

```
35                                      {
36     object->position = add(object->position, value);
37 }
```

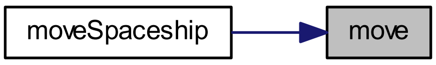
References add(), and _object_t::position.

Referenced by moveSpaceship().

Here is the call graph for this function:



Here is the caller graph for this function:



4.34.1.7 tickObject()

```
void tickObject (
            Object * object,
            uint8_t previousFrameTime )
```

Call the tick function referenced in the object's prototype on the object itself Object might react to the elapsed time. Does nothing when called with NULL.

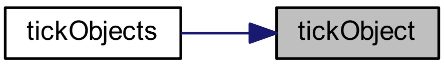
Definition at line 12 of file object.c.

```
                                                      {
    if (object->prototype != NULL) {
        ((TickMethod)pgm_read_word(&object->prototype->tick))(object, previousFrameTime);
    }
}
```

References NULL, _object_t::prototype, and Prototype::tick.

Referenced by tickObjects().

Here is the caller graph for this function:

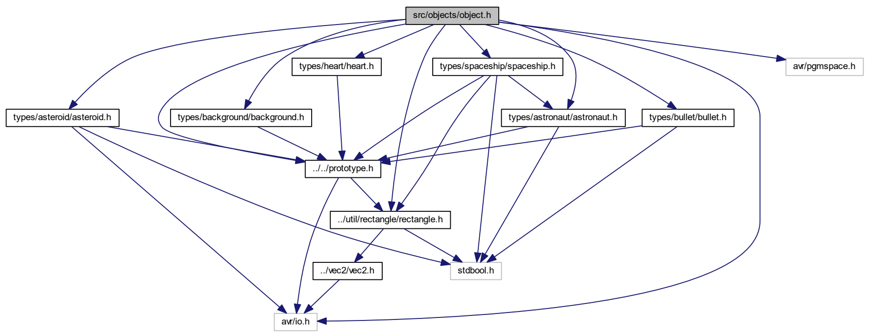


4.35 src/objects/object.h File Reference

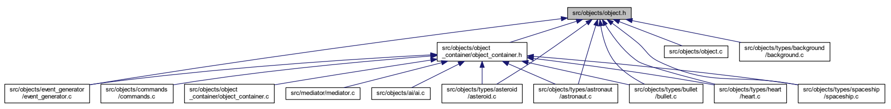
```
#include <avr/io.h>
#include <avr/pgmspace.h>
#include "../util/rectangle/rectangle.h"
#include "types/asteroid/asteroid.h"
#include "types/astronaut/astronaut.h"
#include "types/background/background.h"
```

```
#include "types/spaceship/spaceship.h"
#include "types/bullet/bullet.h"
#include "types/heart/heart.h"
#include "prototype.h"
```

Include dependency graph for object.h:


src/objects/object.h
types/heart/heart.h
types/spaceship/spaceship.h
avr/pgmspace.h
types/asteroid/asteroid.h
types/background/background.h
types/astronaut/astronaut.h
types/bullet/bullet.h
../../prototype.h
../util/rectangle/rectangle.h
../vec2/vec2.h
stdbool.h
avr/io.h

This graph shows which files directly or indirectly include this file:


src/objects/object.h
src/objects/object _container/object_container.h
src/objects/object.c
src/objects/types/background /background.c
src/objects/event_generator /event_generator.c
src/objects/commands /commands.c
src/objects/object _container/object_container.c
src/mediator/mediator.c
src/objects/ai/ai.c
src/objects/types/asteroid /asteroid.c
src/objects/types/astronaut /astronaut.c
src/objects/types/bullet /bullet.c
src/objects/types/heart /heart.c
src/objects/types/spaceship /spaceship.c

Data Structures

- union object_specific_data_t
- struct _object_t

Functions

- Object ∗ createObject (Prototype const ∗prototype, Object ∗holder)
- void tickObject (Object ∗object, uint8_t previousFrameTime)
- void drawObject (Object ∗object, Rectangle compositingWindow)
- Vec2 getSize (Object const ∗object)

  Find out the prototype of the object and return the size of that.
- Vec2 getSizeFromPrototype (Prototype const ∗prototype)
- void move (Object ∗object, Vec2 value)

  Move the position of the object by a vector.
- Rectangle getBoundingBox (Object const ∗object)

4.35.1 Function Documentation

4.35.1.1 createObject()

```
Object* createObject (
            Prototype const * prototype,
            Object * holder )
```

Set the prototype of the holder and initialize all the holder's vale to zero. Return the freshly updated holder

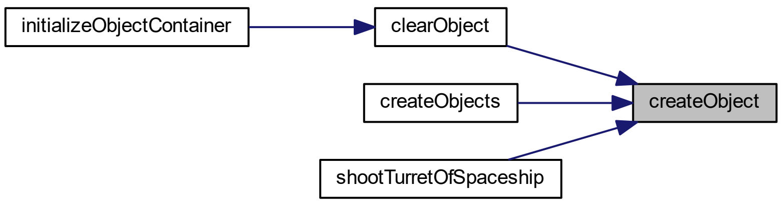
Definition at line 5 of file object.c.

```
                                                        {
    Object empty = {0};
    *holder = empty;
    holder->prototype = prototype;
    return holder;
}
```

References _object_t::prototype.

Referenced by clearObject(), createObjects(), and shootTurretOfSpaceship().

Here is the caller graph for this function:



4.35.1.2 drawObject()

```
void drawObject (
            Object * object,
            Rectangle compositingWindow )
```

Call the draw function referenced in the object's prototype on the object itself. Does nothing when called with NULL.

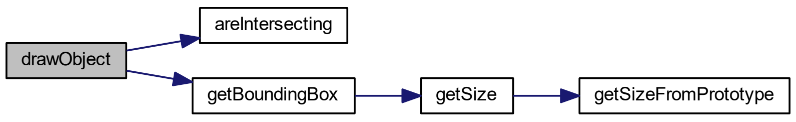
Definition at line 18 of file object.c.

```
                                                     {
    if (object->prototype != NULL && areIntersecting(getBoundingBox(object), compositingWindow)) {
        ((DrawMethod)pgm_read_word(&object->prototype->draw))(object, compositingWindow);
    }
}
```

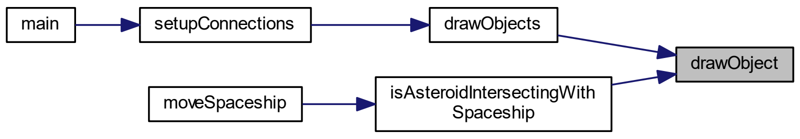
References areIntersecting(), compositingWindow, Prototype::draw, getBoundingBox(), NULL, and _object_t↩ ::prototype.

Referenced by drawObjects(), and isAsteroidIntersectingWithSpaceship().

Here is the call graph for this function:



Here is the caller graph for this function:



4.35.1.3 getBoundingBox()

```
Rectangle getBoundingBox (
              Object const * object )
```

Get a new rectangle from the objects position and its prototype's size

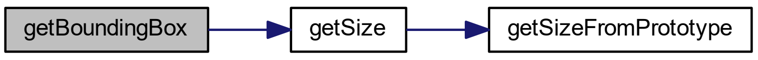
Definition at line 39 of file object.c.

```
                                      {
    return (Rectangle){object->position, getSize(object)};
}
```

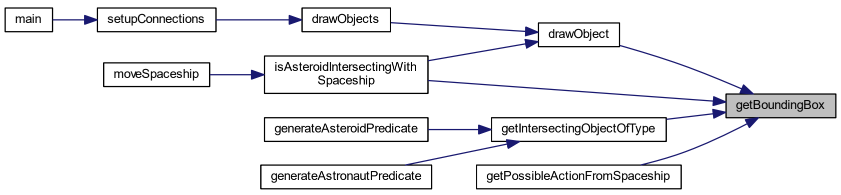
References getSize(), and Rectangle::position.

Referenced by drawObject(), getIntersectingObjectOfType(), getPossibleActionFromSpaceship(), and isAsteroid↩ IntersectingWithSpaceship().

Here is the call graph for this function:



Here is the caller graph for this function:


main
setupConnections
drawObjects
drawObject
moveSpaceship
isAsteroidIntersectingWith Spaceship
getBoundingBox
generateAsteroidPredicate
getIntersectingObjectOfType
generateAstronautPredicate
getPossibleActionFromSpaceship

4.35.1.4 getSize()

```
Vec2 getSize (
            Object const * object )
```

Find out the prototype of the object and return the size of that.

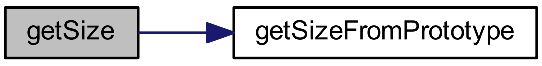
Definition at line 31 of file object.c.

```
31                          {
32     return getSizeFromPrototype(object->prototype);
33 }
```

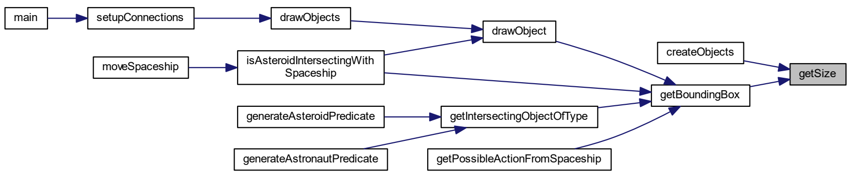
References getSizeFromPrototype(), and _object_t::prototype.

Referenced by createObjects(), and getBoundingBox().

Here is the call graph for this function:



Here is the caller graph for this function:


main
setupConnections
drawObjects
drawObject
moveSpaceship
isAsteroidIntersectingWith Spaceship
createObjects
getSize
getBoundingBox
generateAsteroidPredicate
getIntersectingObjectOfType
generateAstronautPredicate
getPossibleActionFromSpaceship

4.35.1.5 getSizeFromPrototype()

```
Vec2 getSizeFromPrototype (
            Prototype const * prototype )
```

required for casting

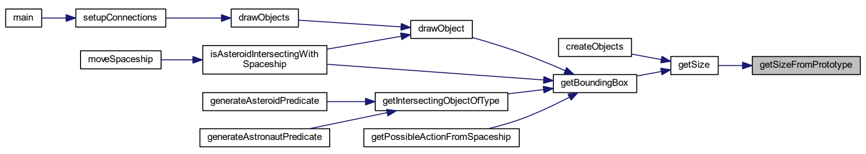
Definition at line 24 of file object.c.

```
24                                                  {
25     /// required for casting
26     uint16_t read = pgm_read_word(&prototype->size);
27     Vec2* v = (Vec2*) &read;
28     return *v;
29 }
```

References Prototype::size.

Referenced by getSize().

Here is the caller graph for this function:


main
setupConnections
drawObjects
drawObject
moveSpaceship
isAsteroidIntersectingWith Spaceship
createObjects
getBoundingBox
getSize
getSizeFromPrototype
generateAsteroidPredicate
getIntersectingObjectOfType
generateAstronautPredicate
getPossibleActionFromSpaceship

4.35.1.6 move()

```
void move (
            Object * object,
            Vec2 value )
```

Move the position of the object by a vector.

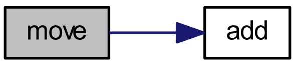
Definition at line 35 of file object.c.

```
35                                    {
36     object->position = add(object->position, value);
37 }
```

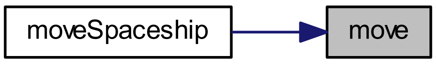
References add(), and _object_t::position.

Referenced by moveSpaceship().

Here is the call graph for this function:



Here is the caller graph for this function:



4.35.1.7 tickObject()

```
void tickObject (
            Object * object,
            uint8_t previousFrameTime )
```

Call the tick function referenced in the object's prototype on the object itself Object might react to the elapsed time. Does nothing when called with NULL.

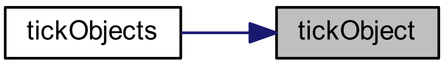
Definition at line 12 of file object.c.

```
                                                         {
    if (object->prototype != NULL) {
        ((TickMethod)pgm_read_word(&object->prototype->tick))(object, previousFrameTime);
    }
}
```

References NULL, _object_t::prototype, and Prototype::tick.

Referenced by tickObjects().

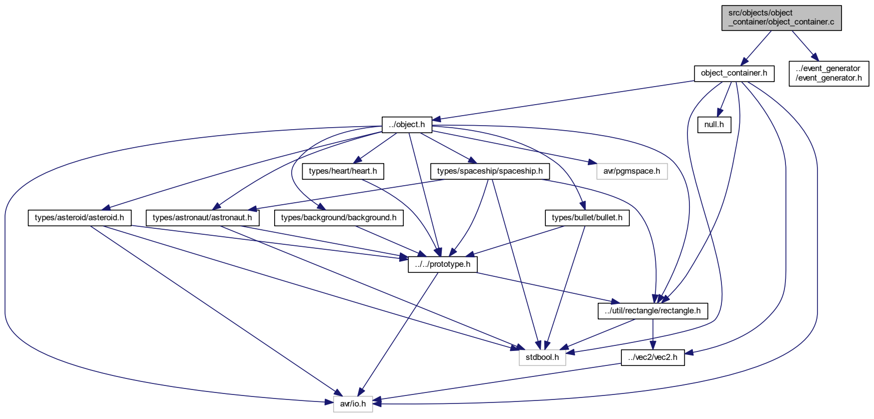
Here is the caller graph for this function:



4.36 src/objects/object_container/object_container.c File Reference

```
#include "object_container.h"
#include "../event_generator/event_generator.h"
```

Include dependency graph for object_container.c:



Functions

- Object ∗ getFirstOfType (Prototype const ∗type)

  may return NULL
- uint8_t getCountOf (Prototype const ∗type)

  Return the number of objects with a prototype being type.
- Object ∗ getIntersectingObjectOfType (Rectangle boundingBox, Prototype const ∗type)
- void clearObject (Object ∗object)
- void drawObjects (Rectangle window)

  Call the draw method of every object.
- void tickObjects (uint8_t previousFrameTime)
- void initializeObjectContainer ()

  Delete every object inside of objects.

4.36.1 Function Documentation

4.36.1.1 clearObject()

```
void clearObject (
            Object ∗ object )
```

Delete the object given by its address from objects It achieves this by setting the object's prototype to NULL.

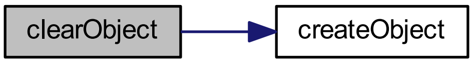
Definition at line 37 of file object_container.c.

```
37                              {
38     createObject(NULL, object);
39 }
```

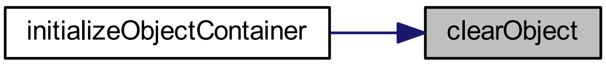
References createObject(), and NULL.

Referenced by initializeObjectContainer().

Here is the call graph for this function:



Here is the caller graph for this function:



4.36.1.2 drawObjects()

```
void drawObjects (
              Rectangle window )
```

Call the draw method of every object.

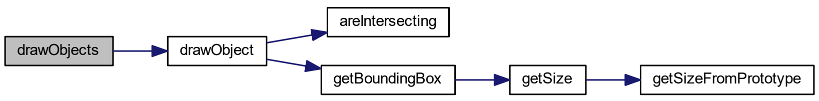
Definition at line 41 of file object_container.c.

```
                                        {
    for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
        drawObject(objects + i, window);
    }
}
```

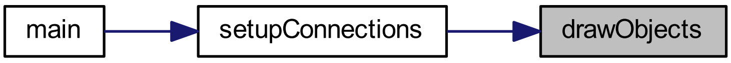
References drawObject(), OBJECT_COUNT, and objects.

Referenced by setupConnections().

Here is the call graph for this function:



Here is the caller graph for this function:



4.36.1.3 getCountOf()

```
uint8_t getCountOf (
            Prototype const * type )
```

Return the number of objects with a prototype being type.

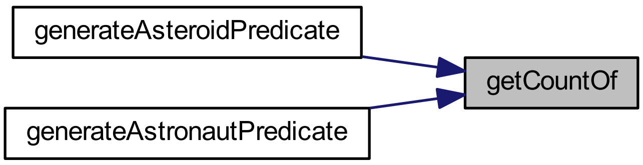
Definition at line 16 of file object_container.c.

```
                                            {
    uint8_t count = 0;
    for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
        if (objects[i].prototype == type) {
            count++;
        }
    }

    return count;
}
```

References OBJECT_COUNT, and objects.

Referenced by generateAsteroidPredicate(), and generateAstronautPredicate().

Here is the caller graph for this function:



4.36.1.4 getFirstOfType()

```
Object* getFirstOfType (
            Prototype const * type )
```

may return NULL

Definition at line 6 of file object_container.c.

```
                                         {
    for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
        if (objects[i].prototype == type) {
            return objects + i;
        }
    }

    return NULL;
}
```

References NULL, OBJECT_COUNT, and objects.

4.36.1.5 getIntersectingObjectOfType()

```
Object* getIntersectingObjectOfType (
            Rectangle boundingBox,
            Prototype const * type )
```

Return a reference to a random object intersecting boundingBox and having a prototype of type

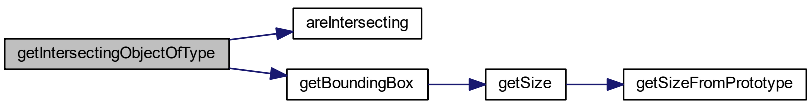
Definition at line 27 of file object_container.c.

```
                                                                              {
    for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
        if (objects[i].prototype == type && areIntersecting(boundingBox, getBoundingBox(objects + i))) {
            return objects + i;
        }
    }

    return NULL;
}
```

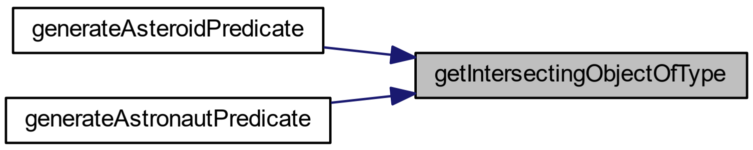
References areIntersecting(), getBoundingBox(), NULL, OBJECT_COUNT, and objects.

Referenced by generateAsteroidPredicate(), and generateAstronautPredicate().

Here is the call graph for this function:



Here is the caller graph for this function:


generateAsteroidPredicate
generateAstronautPredicate
getIntersectingObjectOfType

4.36.1.6 initializeObjectContainer()

```
void initializeObjectContainer ( )
```

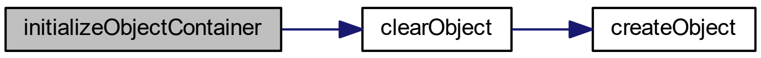
Delete every object inside of objects.

Definition at line 53 of file object_container.c.

```
                              {
    for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
        clearObject(objects + i);
    }
}
```

References clearObject(), OBJECT_COUNT, and objects.

Here is the call graph for this function:



4.36.1.7 tickObjects()

```
void tickObjects (
            uint8_t previousFrameTime )
```

Call the tick method of every object objects might respond to the elapsed time

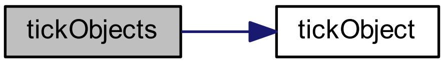
Definition at line 47 of file object_container.c.

```
                                            {
    for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
        tickObject(objects + i, previousFrameTime);
    }
```

```
51 }
```

References OBJECT_COUNT, objects, and tickObject().

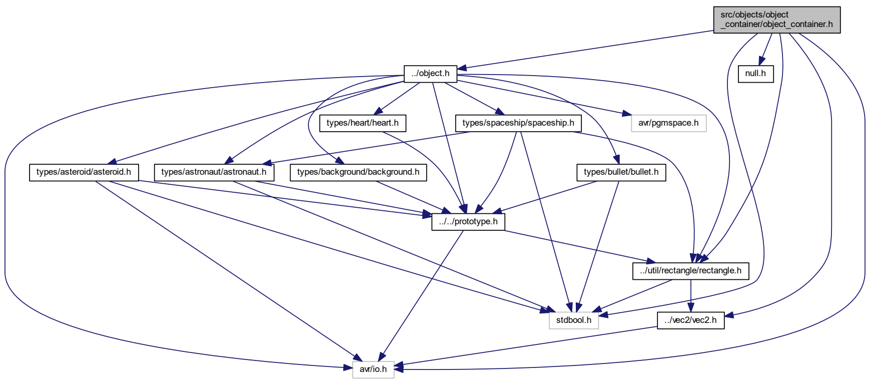
Here is the call graph for this function:



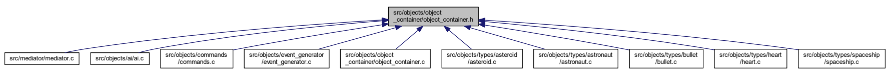
4.37 src/objects/object_container/object_container.h File Reference

```
#include <stdbool.h>
#include <avr/io.h>
#include "null.h"
#include "../object.h"
#include "../../util/rectangle/rectangle.h"
#include "../../util/vec2/vec2.h"
```

Include dependency graph for object_container.h:


src/objects/object
_container/object_container.h
../object.h
null.h
types/heart/heart.h
types/spaceship/spaceship.h
avr/pgmspace.h
types/asteroid/asteroid.h
types/astronaut/astronaut.h
types/background/background.h
types/bullet/bullet.h
../../prototype.h
../util/rectangle/rectangle.h
stdbool.h
../vec2/vec2.h
avr/io.h

This graph shows which files directly or indirectly include this file:



Macros

- #define OBJECT_COUNT 10
- #define BACKGROUND_INDEX 0
- #define SPACESHIP_INDEX 1
- #define CHARACTER_INDEX 2
- #define spaceshipObject (objects + SPACESHIP_INDEX)
- #define character (objects + CHARACTER_INDEX)
- #define getEmptyObjectSpace() getFirstOfType(NULL)

  may return NULL

Functions

- Object ∗ getFirstOfType (Prototype const ∗type)

  may return NULL
- uint8_t getCountOf (Prototype const ∗type)

  Return the number of objects with a prototype being type.
- Object ∗ getIntersectingObjectOfType (Rectangle boundingBox, Prototype const ∗type)
- void tickObjects (uint8_t previousFrameTime)
- void drawObjects (Rectangle window)

  Call the draw method of every object.
- void clearObject (Object ∗object)
- void initializeObjectContainer ()

  Delete every object inside of objects.

Variables

- Object objects [OBJECT_COUNT]

  The actual container.

4.37.1 Macro Definition Documentation

4.37.1.1 BACKGROUND_INDEX

```
#define BACKGROUND_INDEX 0
```

Definition at line 22 of file object_container.h.

4.37.1.2 character

```
#define character (objects + CHARACTER_INDEX)
```

Definition at line 27 of file object_container.h.

4.37.1.3 CHARACTER_INDEX

```
#define CHARACTER_INDEX 2
```

Definition at line 24 of file object_container.h.

4.37.1.4 getEmptyObjectSpace

```
#define getEmptyObjectSpace( ) getFirstOfType(NULL)
```

may return NULL

Definition at line 36 of file object_container.h.

4.37.1.5 OBJECT_COUNT

```
#define OBJECT_COUNT 10
```

Contain up to OBJECT_COUNT objects. Provide the basic functionality to access, search and modify these objects. For ease of use, there are some convenience global variables for accessing objects that are very commonly accessed.

Definition at line 21 of file object_container.h.

4.37.1.6 SPACESHIP_INDEX

```
#define SPACESHIP_INDEX 1
```

Definition at line 23 of file object_container.h.

4.37.1.7 spaceshipObject

```
#define spaceshipObject (objects + SPACESHIP_INDEX)
```

Definition at line 26 of file object_container.h.

4.37.2 Function Documentation

4.37.2.1 clearObject()

```
void clearObject (
            Object * object )
```

Delete the object given by its address from objects It achieves this by setting the object's prototype to NULL.

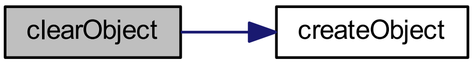
Definition at line 37 of file object_container.c.

```
                                       {
    createObject(NULL, object);
}
```

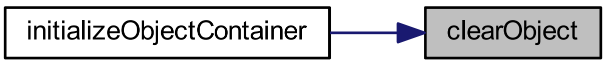
References createObject(), and NULL.

Referenced by initializeObjectContainer().

Here is the call graph for this function:



Here is the caller graph for this function:



4.37.2.2 drawObjects()

```
void drawObjects (
            Rectangle window )
```

Call the draw method of every object.

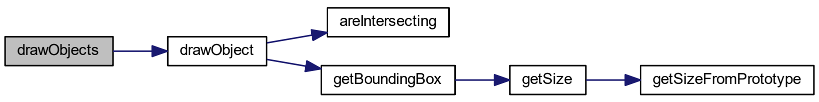
Definition at line 41 of file object_container.c.

```
                                        {
    for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
        drawObject(objects + i, window);
    }
}
```

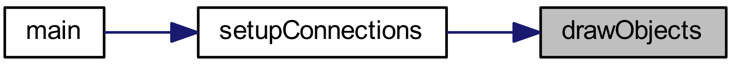
References drawObject(), OBJECT_COUNT, and objects.

Referenced by setupConnections().

Here is the call graph for this function:



Here is the caller graph for this function:



4.37.2.3 getCountOf()

```
uint8_t getCountOf (
            Prototype const * type )
```

Return the number of objects with a prototype being type.

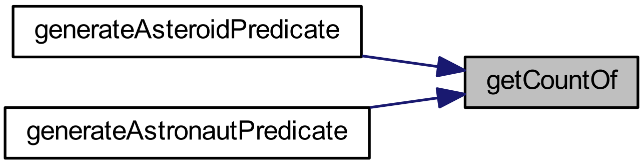
Definition at line 16 of file object_container.c.

```
                                              {
    uint8_t count = 0;
    for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
        if (objects[i].prototype == type) {
            count++;
        }
    }

    return count;
}
```

References OBJECT_COUNT, and objects.

Referenced by generateAsteroidPredicate(), and generateAstronautPredicate().

Here is the caller graph for this function:



4.37.2.4 getFirstOfType()

```
Object* getFirstOfType (
            Prototype const * type )
```

may return NULL

Definition at line 6 of file object_container.c.

```
                                                {
    for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
        if (objects[i].prototype == type) {
            return objects + i;
        }
    }

    return NULL;
}
```

References NULL, OBJECT_COUNT, and objects.

4.37.2.5 getIntersectingObjectOfType()

```
Object* getIntersectingObjectOfType (
            Rectangle boundingBox,
            Prototype const * type )
```

Return a reference to a random object intersecting boundingBox and having a prototype of type

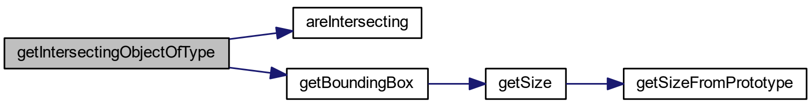
Definition at line 27 of file object_container.c.

```
                                                                                {
    for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
        if (objects[i].prototype == type && areIntersecting(boundingBox, getBoundingBox(objects + i))) {
            return objects + i;
        }
    }

    return NULL;
}
```

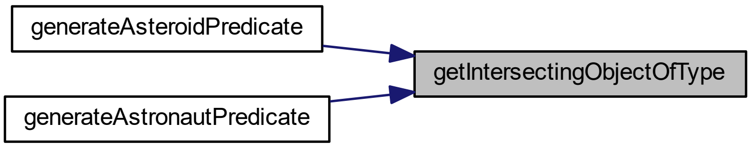
References areIntersecting(), getBoundingBox(), NULL, OBJECT_COUNT, and objects.

Referenced by generateAsteroidPredicate(), and generateAstronautPredicate().

Here is the call graph for this function:



Here is the caller graph for this function:


generateAsteroidPredicate
generateAstronautPredicate
getIntersectingObjectOfType

4.37.2.6 initializeObjectContainer()

```
void initializeObjectContainer ( )
```

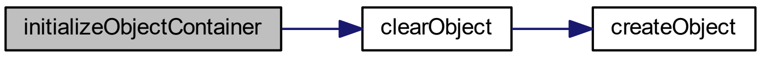
Delete every object inside of objects.

Definition at line 53 of file object_container.c.

```
                                 {
    for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
        clearObject(objects + i);
    }
}
```

References clearObject(), OBJECT_COUNT, and objects.

Here is the call graph for this function:



4.37.2.7 tickObjects()

```
void tickObjects (
            uint8_t previousFrameTime )
```

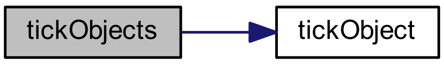
Call the tick method of every object objects might respond to the elapsed time

Definition at line 47 of file object_container.c.

```
                                            {
    for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
        tickObject(objects + i, previousFrameTime);
    }
}
```

References OBJECT_COUNT, objects, and tickObject().

Here is the call graph for this function:



4.37.3 Variable Documentation

4.37.3.1 objects

```
Object objects[OBJECT_COUNT]
```

The actual container.

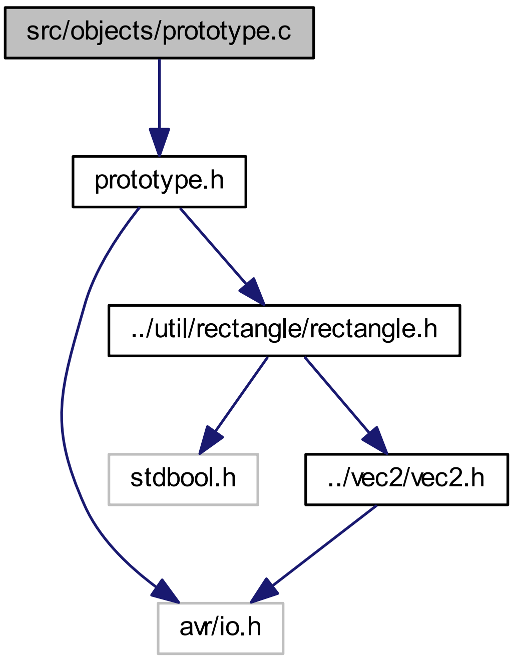
Definition at line 29 of file object_container.h.

Referenced by drawObjects(), getCountOf(), getFirstOfType(), getIntersectingObjectOfType(), initializeObject↩ Container(), moveSpaceship(), and tickObjects().

4.38 src/objects/prototype.c File Reference

```
#include "prototype.h"
```

Include dependency graph for prototype.c:



Functions

- void tickObjectFromPrototype (Object ∗object)
- void drawObjectFromPrototype (Object ∗object)

Variables

- Prototype temp

4.38.1 Function Documentation

4.38.1.1 drawObjectFromPrototype()

```
void drawObjectFromPrototype (
            Object ∗ object )
```

Definition at line 19 of file prototype.c.

```
                                              {
    /*loadPrototype(object->prototype);
    temp.draw(object);*/
}
```

4.38.1.2 tickObjectFromPrototype()

```
void tickObjectFromPrototype (
             Object * object )
```

Definition at line 14 of file prototype.c.

```
                                              {
    loadPrototype(object->prototype);
    temp.tick(object);
}
```

4.38.2 Variable Documentation

4.38.2.1 temp

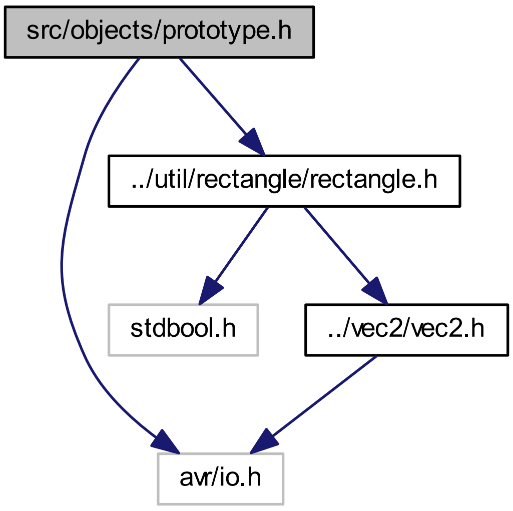
```
Prototype temp
```

Definition at line 4 of file prototype.c.

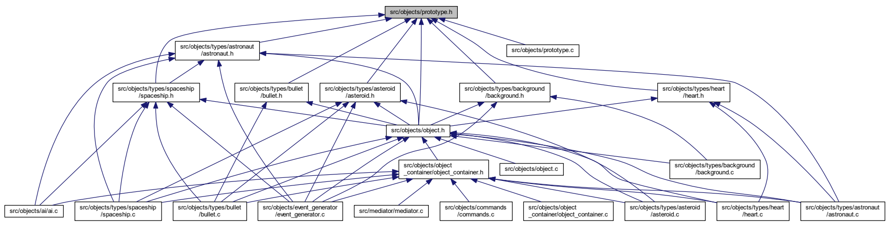
4.39 src/objects/prototype.h File Reference

```
#include <avr/io.h>
#include "../util/rectangle/rectangle.h"
```

Include dependency graph for prototype.h:



This graph shows which files directly or indirectly include this file:


src/objects/prototype.h
src/objects/types/astronaut/astronaut.h
src/objects/prototype.c
src/objects/types/spaceship/spaceship.h
src/objects/types/bullet/bullet.h
src/objects/types/asteroid/asteroid.h
src/objects/types/background/background.h
src/objects/types/heart/heart.h
src/objects/object.h
src/objects/object_container/object_container.h
src/objects/object.c
src/objects/types/background/background.c
src/objects/ai/ai.c
src/objects/types/spaceship/spaceship.c
src/objects/types/bullet/bullet.c
src/objects/event_generator/event_generator.c
src/mediator/mediator.c
src/objects/commands/commands.c
src/objects/object_container/object_container.c
src/objects/types/asteroid/asteroid.c
src/objects/types/heart/heart.c
src/objects/types/astronaut/astronaut.c

Data Structures

- struct Prototype

Typedefs

- typedef struct _object_t Object
- typedef void(∗ TickMethod) (Object ∗, uint8_t)
- typedef void(∗ DrawMethod) (Object ∗, Rectangle)

  Draw the given object if its overlapping with the given rectangle.

4.39.1 Typedef Documentation

4.39.1.1 DrawMethod

```
typedef void(∗ DrawMethod) (Object ∗, Rectangle)
```

Draw the given object if its overlapping with the given rectangle.

Definition at line 19 of file prototype.h.

4.39.1.2 Object

```
typedef struct _object_t Object
```

Definition at line 1 of file prototype.h.

4.39.1.3 TickMethod

```
typedef void(* TickMethod) (Object *, uint8_t)
```

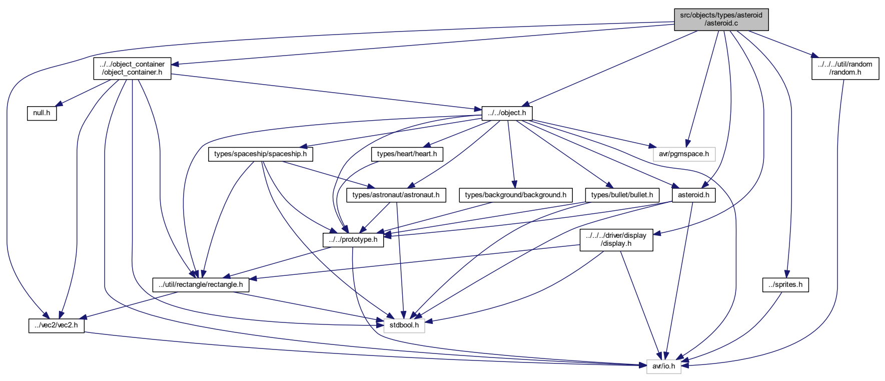
Update the inner state of the given object The first argument is the object itself, the second is the elapsed time in milliseconds.

Definition at line 16 of file prototype.h.

4.40 src/objects/types/asteroid/asteroid.c File Reference

```
#include "asteroid.h"
#include <avr/pgmspace.h>
#include "../sprites.h"
#include "../../object.h"
#include "../../object_container/object_container.h"
#include "../../../util/vec2/vec2.h"
#include "../../../util/random/random.h"
#include "../../../driver/display/display.h"
```

Include dependency graph for asteroid.c:



Functions

- bool mineAsteroid (Object *asteroid)
- bool isAsteroidIntersectingWithSpaceship (Object *asteroid, Object *spaceship)

Variables

- const Prototype Asteroid PROGMEM

4.40.1 Function Documentation

4.40.1.1 isAsteroidIntersectingWithSpaceship()

```
bool isAsteroidIntersectingWithSpaceship (
            Object * asteroid,
            Object * spaceship )
```

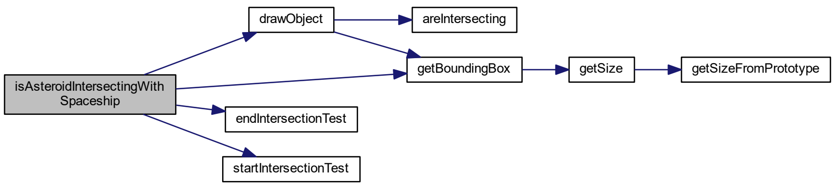
Definition at line 31 of file asteroid.c.

```
                                                                                        {
    Rectangle bb = getBoundingBox(asteroid);
    startIntersectionTest(bb);
    drawObject(asteroid, bb);
    drawObject(spaceship, bb);
    return endIntersectionTest();
}
```

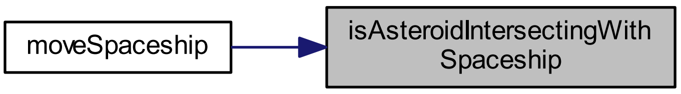
References drawObject(), endIntersectionTest(), getBoundingBox(), and startIntersectionTest().

Referenced by moveSpaceship().

Here is the call graph for this function:



Here is the caller graph for this function:



4.40.1.2 mineAsteroid()

```
bool mineAsteroid (
            Object * asteroid )
```

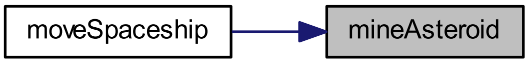
Definition at line 13 of file asteroid.c.

```
                                        {
    return ++asteroid->as.asteroid.animationFrame == IDLE_FRAME_COUNT;
}
```

References _asteroid_t::animationFrame, _object_t::as, object_specific_data_t::asteroid, and IDLE_FRAME_CO↩
UNT.

Referenced by moveSpaceship().

Here is the caller graph for this function:



4.40.2 Variable Documentation

4.40.2.1 PROGMEM

```
const Prototype Asteroid PROGMEM
```

Initial value:
```
= {
    .tick = tick,
    .draw = draw,
    .size = ASTEROID_SIZE,
}
```

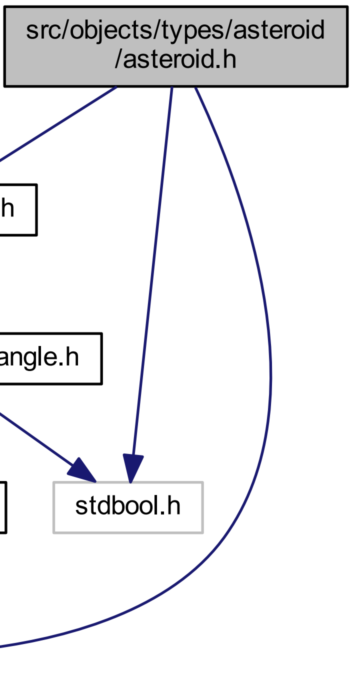
Definition at line 47 of file asteroid.c.

4.41 src/objects/types/asteroid/asteroid.h File Reference

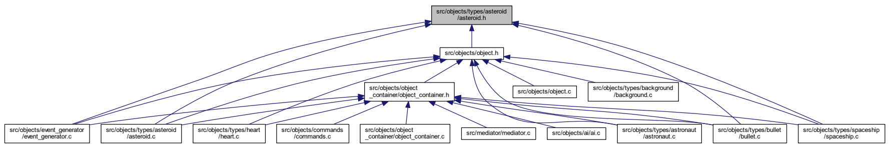
```
#include <avr/io.h>
#include <stdbool.h>
```

```
#include "../../prototype.h"
```

Include dependency graph for asteroid.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct _asteroid_t

Macros

- #define ASTEROID_SIZE ((Vec2){8, 8})
- #define IDLE_FRAME_COUNT 4
- #define EXPLODING_FRAME_COUNT 3
- #define EXPLODING_FRAME_CHANGE_INTERVAL 3

Functions

- bool mineAsteroid (Object ∗asteroid)
- bool isAsteroidIntersectingWithSpaceship (Object ∗asteroid, Object ∗spaceship)

Variables

- const Prototype Asteroid

4.41.1 Macro Definition Documentation

4.41.1.1 ASTEROID_SIZE

```
#define ASTEROID_SIZE ((Vec2){8, 8})
```

Definition at line 10 of file asteroid.h.

4.41.1.2 EXPLODING_FRAME_CHANGE_INTERVAL

```
#define EXPLODING_FRAME_CHANGE_INTERVAL 3
```

Definition at line 13 of file asteroid.h.

4.41.1.3 EXPLODING_FRAME_COUNT

```
#define EXPLODING_FRAME_COUNT 3
```

Definition at line 12 of file asteroid.h.

4.41.1.4 IDLE_FRAME_COUNT

```
#define IDLE_FRAME_COUNT 4
```

Definition at line 11 of file asteroid.h.

4.41.2 Function Documentation

4.41.2.1 isAsteroidIntersectingWithSpaceship()

```
bool isAsteroidIntersectingWithSpaceship (
             Object * asteroid,
             Object * spaceship )
```

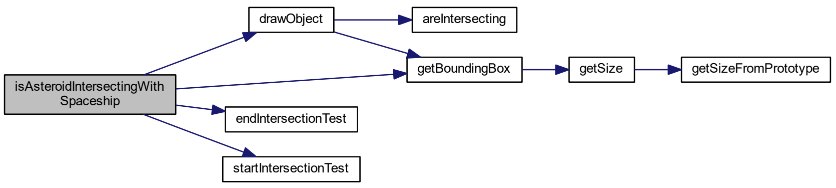
Definition at line 31 of file asteroid.c.

```
                                                                              {
      Rectangle bb = getBoundingBox(asteroid);
      startIntersectionTest(bb);
      drawObject(asteroid, bb);
      drawObject(spaceship, bb);
      return endIntersectionTest();
  }
```

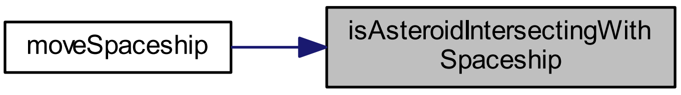
References drawObject(), endIntersectionTest(), getBoundingBox(), and startIntersectionTest().

Referenced by moveSpaceship().

Here is the call graph for this function:



Here is the caller graph for this function:



4.41.2.2 mineAsteroid()

```
bool mineAsteroid (
             Object * asteroid )
```

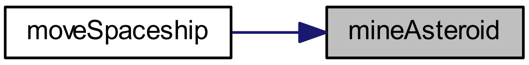
Definition at line 13 of file asteroid.c.

```
                                      {
      return ++asteroid->as.asteroid.animationFrame == IDLE_FRAME_COUNT;
  }
```

References _asteroid_t::animationFrame, _object_t::as, object_specific_data_t::asteroid, and IDLE_FRAME_CO↩ UNT.

Referenced by moveSpaceship().

Here is the caller graph for this function:



4.41.3 Variable Documentation

4.41.3.1 Asteroid

```
const Prototype Asteroid
```

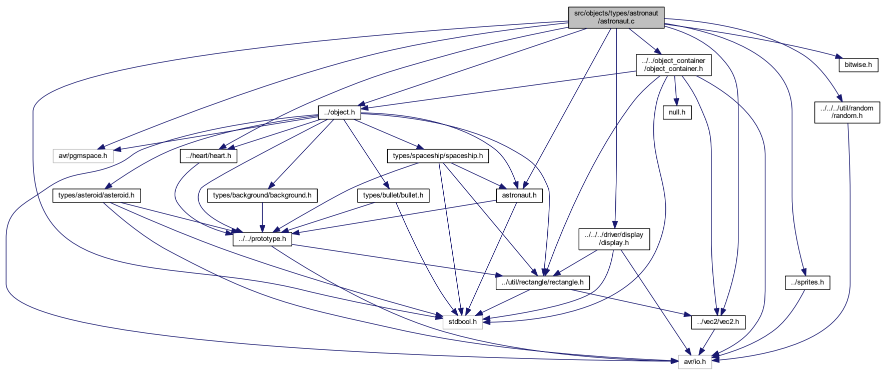
Definition at line 14 of file asteroid.h.

Referenced by generateAsteroidPredicate(), and moveSpaceship().

4.42 src/objects/types/astronaut/astronaut.c File Reference

```
#include "astronaut.h"
#include <stdbool.h>
#include <avr/pgmspace.h>
#include "bitwise.h"
#include "../../../driver/display/display.h"
#include "../heart/heart.h"
#include "../../object_container/object_container.h"
#include "../sprites.h"
#include "../../../util/vec2/vec2.h"
#include "../../../util/random/random.h"
#include "../../object.h"
```

Include dependency graph for astronaut.c:


src/objects/types/astronaut/astronaut.c
../../object_container/object_container.h
bitwise.h
../../../util/random/random.h
null.h
../object.h
avr/pgmspace.h
../heart/heart.h
types/spaceship/spaceship.h
types/asteroid/asteroid.h
types/background/background.h
types/bullet/bullet.h
astronaut.h
../../prototype.h
../../../driver/display/display.h
../util/rectangle/rectangle.h
../sprites.h
stdbool.h
../vec2/vec2.h
avr/io.h

Macros

- #define IS_MIRRORED_BIT 0
- #define IS_CONTROLLING_SPACESHIP_BIT 1
- #define WAS_DOING_ACTION_BIT 2

Functions

- bool getIsControllingSpaceship (Object ∗astronaut)
- void moveAstronaut (Object ∗astronaut, Vec2 direction)
- void makeAstronautDoAction (Object ∗astronaut)

Variables

- const Prototype Astronaut PROGMEM

4.42.1 Macro Definition Documentation

4.42.1.1 IS_CONTROLLING_SPACESHIP_BIT

```
#define IS_CONTROLLING_SPACESHIP_BIT 1
```

Definition at line 17 of file astronaut.c.

4.42.1.2 IS_MIRRORED_BIT

```
#define IS_MIRRORED_BIT 0
```

Definition at line 16 of file astronaut.c.

4.42.1.3 WAS_DOING_ACTION_BIT

```
#define WAS_DOING_ACTION_BIT 2
```

Definition at line 18 of file astronaut.c.

4.42.2 Function Documentation

4.42.2.1 getIsControllingSpaceship()

```
bool getIsControllingSpaceship (
            Object * astronaut )
```

Definition at line 36 of file astronaut.c.

```
                                                  {
    return getBit(astronaut->as.astronaut.flags, IS_CONTROLLING_SPACESHIP_BIT);
}
```

References _object_t::as, object_specific_data_t::astronaut, _astronaut_t::flags, getBit, and IS_CONTROLLING_SPACESHIP_BIT.

4.42.2.2 makeAstronautDoAction()

```
void makeAstronautDoAction (
            Object * astronaut )
```

Definition at line 87 of file astronaut.c.

```
                                              {
    if (
    astronaut->as.astronaut.timeSinceLastAction < TIME_BETWEEN_ACTION_CHANGE
    && getWasDoingAction(astronaut)
    ) {
        return;
    }
    astronaut->as.astronaut.timeSinceLastAction = 0;
    setWasDoingAction(astronaut, true);

    if (getIsControllingSpaceship(astronaut)) {
        setIsControllingSpaceship(astronaut, false);
        } else {
         Object* heart;
         switch (getPossibleActionFromSpaceship(astronaut)) {
             case shootTurret:
             shootTurretOfSpaceship();
             break;
             case showLove:
             heart = getEmptyObjectSpace();
             if (heart != NULL) {
                 for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
                     if (
                     objects[i].prototype == &Astronaut &&
                     areIntersecting(getBoundingBox(astronaut), getBoundingBox(objects + i)) &&
                     objects + i != astronaut
                     ) {
                         createObject(&Heart, heart);
                         heart->position = add(astronaut->position, (Vec2){(getRandomNumber() % 11) - 5,
    -(getRandomNumber() % 5) - 10});
                         break;
                     }
                 }
             }
             break;
             case repairingSpaceship:
             if (spaceshipObject->as.spaceship.healthLoss > 0) {
                 spaceshipObject->as.spaceship.healthLoss--;
             }
             break;
             case controllingSpaceship:
             setIsControllingSpaceship(astronaut, true);
             break;
             default:
             break;
         }
     }
}
```

References _object_t::as, object_specific_data_t::astronaut, TIME_BETWEEN_ACTION_CHANGE, and _astronaut_t::timeSinceLastAction.

4.42.2.3 moveAstronaut()

```
void moveAstronaut (
            Object * astronaut,
            Vec2 direction )
```

Definition at line 58 of file astronaut.c.

```
                                                       {
    if (
    astronaut->as.astronaut.timeSinceLastAction < TIME_BETWEEN_ACTION_CHANGE
    && !getWasDoingAction(astronaut)
    ) {
        return;
    }
    astronaut->as.astronaut.timeSinceLastAction = 0;
    setWasDoingAction(astronaut, false);

    if (getIsControllingSpaceship(astronaut)) {
        moveSpaceship((Vec2){direction.x, 0});
        moveSpaceship((Vec2){0, direction.y});
    } else {
        Vec2 proposedPosition = add(astronaut->position, direction);
        Rectangle proposedBoundingBox = (Rectangle){proposedPosition, getSize(astronaut)};
        if (isOnboard(proposedBoundingBox)) {
            astronaut->position = proposedPosition;
            astronaut->as.astronaut.animationFrame = (astronaut->as.astronaut.animationFrame + 1) %
    MOVE_FRAME_COUNT;
        }

        if (direction.x == 0) {
            astronaut->as.astronaut.animationFrame = 0;
        }

        setIsMirrored(astronaut, direction.x < 0);
    }
}
```

References _object_t::as, object_specific_data_t::astronaut, TIME_BETWEEN_ACTION_CHANGE, and _↩
astronaut_t::timeSinceLastAction.

4.42.3 Variable Documentation

4.42.3.1 PROGMEM

```
const Prototype Astronaut PROGMEM
```

Initial value:

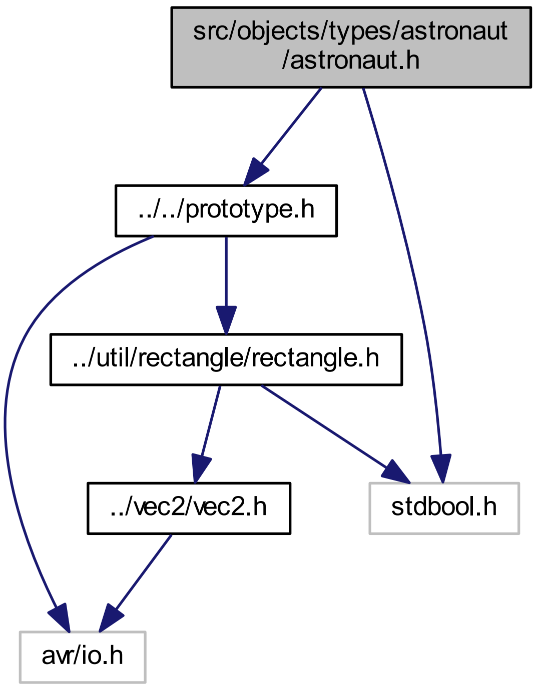
```
= {
    .tick = tick,
    .draw = draw,
    .size = ASTRONAUT_SIZE
}
```

Definition at line 143 of file astronaut.c.

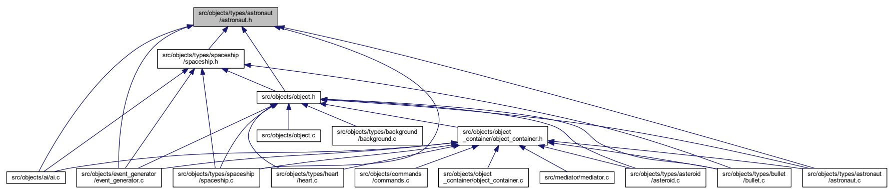
4.43 src/objects/types/astronaut/astronaut.h File Reference

```
#include "../../prototype.h"
#include <stdbool.h>
```

Include dependency graph for astronaut.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct _astronaut_t

Macros

- #define ASTRONAUT_SIZE ((Vec2){5, 5})
- #define MOVE_FRAME_COUNT 4
- #define TIME_BETWEEN_ACTION_CHANGE 40

Enumerations

- enum Action {
  noAction = 0, controllingSpaceship, shootTurret, showLove,
  repairingSpaceship, ACTION_COUNT }

Functions

- void moveAstronaut (Object ∗astronaut, Vec2 unitVector)
- void makeAstronautDoAction (Object ∗astronaut)
- bool getIsControllingSpaceship (Object ∗astronaut)

Variables

- const Prototype Astronaut

4.43.1 Macro Definition Documentation

4.43.1.1 ASTRONAUT_SIZE

```
#define ASTRONAUT_SIZE ((Vec2){5, 5})
```

Definition at line 9 of file astronaut.h.

4.43.1.2 MOVE_FRAME_COUNT

```
#define MOVE_FRAME_COUNT 4
```

Definition at line 10 of file astronaut.h.

4.43.1.3 TIME_BETWEEN_ACTION_CHANGE

```
#define TIME_BETWEEN_ACTION_CHANGE 40
```

Between two consecutive actions (or movements) there has to be at least this many milliseconds

Definition at line 14 of file astronaut.h.

4.43.2 Enumeration Type Documentation

4.43.2.1 Action

```
enum Action
```

Enumerator

noAction	
controllingSpaceship	
shootTurret	
showLove	
repairingSpaceship	
ACTION_COUNT	

Definition at line 15 of file astronaut.h.

```
              {
    noAction = 0,
    controllingSpaceship,
    shootTurret,
    showLove,
    repairingSpaceship,
    ACTION_COUNT
} Action;
```

4.43.3 Function Documentation

4.43.3.1 getIsControllingSpaceship()

```
bool getIsControllingSpaceship (
            Object * astronaut )
```

Definition at line 36 of file astronaut.c.

```
                                                   {
    return getBit(astronaut->as.astronaut.flags, IS_CONTROLLING_SPACESHIP_BIT);
}
```

References _object_t::as, object_specific_data_t::astronaut, _astronaut_t::flags, getBit, and IS_CONTROLLING↩_SPACESHIP_BIT.

4.43.3.2 makeAstronautDoAction()

```
void makeAstronautDoAction (
            Object * astronaut )
```

Definition at line 87 of file astronaut.c.

```
                                              {
    if (
    astronaut->as.astronaut.timeSinceLastAction < TIME_BETWEEN_ACTION_CHANGE
    && getWasDoingAction(astronaut)
    ) {
        return;
    }
    astronaut->as.astronaut.timeSinceLastAction = 0;
    setWasDoingAction(astronaut, true);

    if (getIsControllingSpaceship(astronaut)) {
        setIsControllingSpaceship(astronaut, false);
        } else {
         Object* heart;
         switch (getPossibleActionFromSpaceship(astronaut)) {
             case shootTurret:
```

```
            shootTurretOfSpaceship();
            break;
            case showLove:
            heart = getEmptyObjectSpace();
            if (heart != NULL) {
                for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
                    if (
                    objects[i].prototype == &Astronaut &&
                    areIntersecting(getBoundingBox(astronaut), getBoundingBox(objects + i)) &&
                    objects + i != astronaut
                    ) {
                        createObject(&Heart, heart);
                        heart->position = add(astronaut->position, (Vec2){(getRandomNumber() % 11) - 5,
        -(getRandomNumber() % 5) - 10});
                        break;
                    }
                }
            }
            break;
            case repairingSpaceship:
            if (spaceshipObject->as.spaceship.healthLoss > 0) {
                spaceshipObject->as.spaceship.healthLoss--;
            }
            break;
            case controllingSpaceship:
            setIsControllingSpaceship(astronaut, true);
            break;
            default:
            break;
        }
    }
}
```

References _object_t::as, object_specific_data_t::astronaut, TIME_BETWEEN_ACTION_CHANGE, and _↩
astronaut_t::timeSinceLastAction.

4.43.3.3 moveAstronaut()

```
void moveAstronaut (
              Object * astronaut,
              Vec2 unitVector )
```

Definition at line 58 of file astronaut.c.

```
                                                      {
    if (
    astronaut->as.astronaut.timeSinceLastAction < TIME_BETWEEN_ACTION_CHANGE
    && !getWasDoingAction(astronaut)
    ) {
        return;
    }
    astronaut->as.astronaut.timeSinceLastAction = 0;
    setWasDoingAction(astronaut, false);

    if (getIsControllingSpaceship(astronaut)) {
        moveSpaceship((Vec2){direction.x, 0});
        moveSpaceship((Vec2){0, direction.y});
    } else {
        Vec2 proposedPosition = add(astronaut->position, direction);
        Rectangle proposedBoundingBox = (Rectangle){proposedPosition, getSize(astronaut)};
        if (isOnboard(proposedBoundingBox)) {
            astronaut->position = proposedPosition;
            astronaut->as.astronaut.animationFrame = (astronaut->as.astronaut.animationFrame + 1) %
    MOVE_FRAME_COUNT;
        }

        if (direction.x == 0) {
            astronaut->as.astronaut.animationFrame = 0;
        }

        setIsMirrored(astronaut, direction.x < 0);
    }
}
```

References _object_t::as, object_specific_data_t::astronaut, TIME_BETWEEN_ACTION_CHANGE, and _↩
astronaut_t::timeSinceLastAction.

4.43.4 Variable Documentation

4.43.4.1 Astronaut

```
const Prototype Astronaut
```

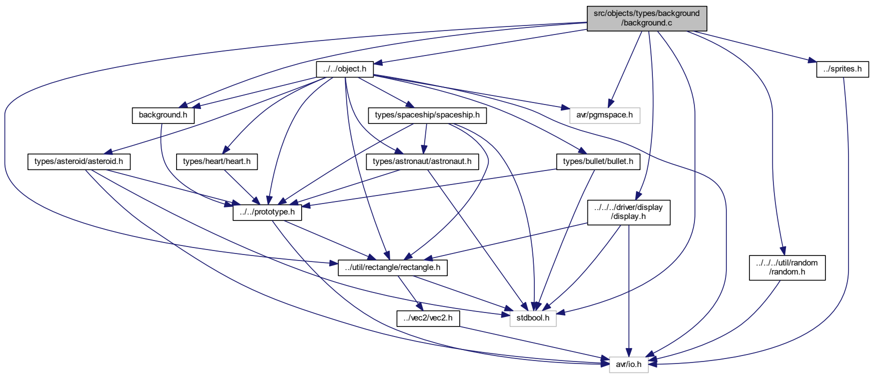
Definition at line 25 of file astronaut.h.

Referenced by createObjects(), generateAstronautPredicate(), and moveSpaceship().

4.44 src/objects/types/background/background.c File Reference

```
#include "background.h"
#include <stdbool.h>
#include <avr/pgmspace.h>
#include "../../object.h"
#include "../../../util/rectangle/rectangle.h"
#include "../../../util/random/random.h"
#include "../sprites.h"
#include "../../../driver/display/display.h"
```

Include dependency graph for background.c:



Data Structures

- struct Star

Functions

- void initializeBackground ()

Variables

- const Prototype Background PROGMEM

4.44.1 Function Documentation

4.44.1.1 initializeBackground()

```
void initializeBackground ( )
```

Definition at line 60 of file background.c.

```
60                          {
61     for (uint8_t i = 0; i < STAR_COUNT; i++) {
62         backgroundStars[i] = createStarOnTheRight();
63         backgroundStars[i].position.x = getRandomNumber();
64     }
65 }
```

References STAR_COUNT.

4.44.2 Variable Documentation

4.44.2.1 PROGMEM

```
const Prototype Background PROGMEM
```

Initial value:

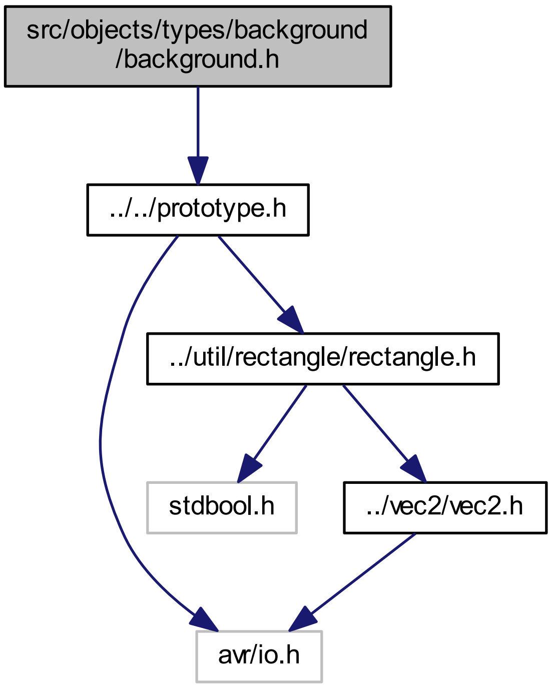
```
= {
    .tick = tick,
    .draw = draw,
    .size = (Vec2){DISPLAY_WIDTH_IN_PIXELS, DISPLAY_HEIGHT_IN_PIXELS}
}
```

Definition at line 67 of file background.c.

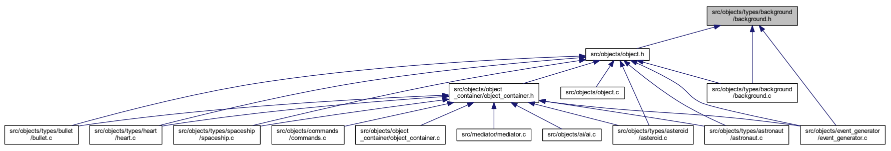
4.45 src/objects/types/background/background.h File Reference

```
#include "../../prototype.h"
```

Include dependency graph for background.h:


src/objects/types/background /background.h
../../prototype.h
../util/rectangle/rectangle.h
stdbool.h
../vec2/vec2.h
avr/io.h

This graph shows which files directly or indirectly include this file:


src/objects/types/background /background.h
src/objects/object.h
src/objects/object _container/object_container.h
src/objects/object.c
src/objects/types/background /background.c
src/objects/types/bullet /bullet.c
src/objects/types/heart /heart.c
src/objects/types/spaceship /spaceship.c
src/objects/commands /commands.c
src/objects/object _container/object_container.c
src/mediator/mediator.c
src/objects/ai/ai.c
src/objects/types/asteroid /asteroid.c
src/objects/types/astronaut /astronaut.c
src/objects/event_generator /event_generator.c

Data Structures

- struct _background_t

Macros

- #define STAR_COUNT 8
- #define STAR_SIZE 3
- #define STAR_SHAPE_COUNT 3

Functions

- void initializeBackground ()

Variables

- const Prototype Background

4.45.1 Macro Definition Documentation

4.45.1.1 STAR_COUNT

```
#define STAR_COUNT 8
```

Definition at line 8 of file background.h.

4.45.1.2 STAR_SHAPE_COUNT

```
#define STAR_SHAPE_COUNT 3
```

Definition at line 10 of file background.h.

4.45.1.3 STAR_SIZE

```
#define STAR_SIZE 3
```

Definition at line 9 of file background.h.

4.45.2 Function Documentation

4.45.2.1 initializeBackground()

```
void initializeBackground ( )
```

Definition at line 60 of file background.c.

```
                                {
    for (uint8_t i = 0; i < STAR_COUNT; i++) {
        backgroundStars[i] = createStarOnTheRight();
        backgroundStars[i].position.x = getRandomNumber();
    }
}
```

References STAR_COUNT.

4.45.3 Variable Documentation

4.45.3.1 Background

```
const Prototype Background
```

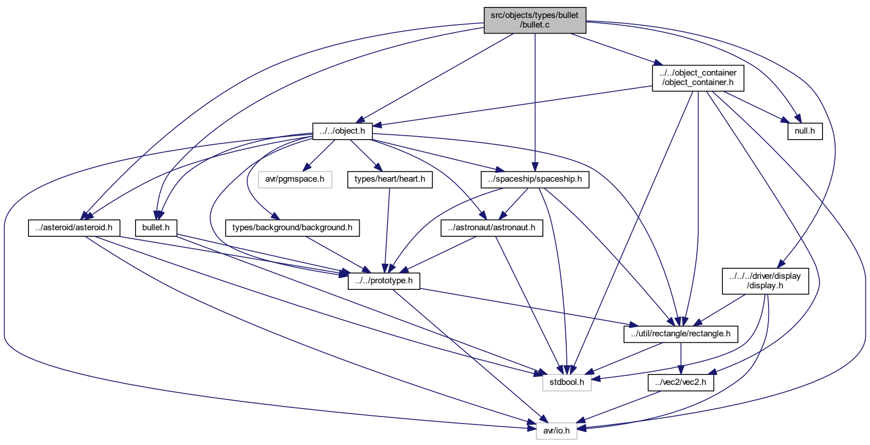
Definition at line 11 of file background.h.

Referenced by createObjects().

4.46 src/objects/types/bullet/bullet.c File Reference

```
#include "bullet.h"
#include "../asteroid/asteroid.h"
#include "../spaceship/spaceship.h"
#include "../../object.h"
#include "../../object_container/object_container.h"
#include "null.h"
#include "../../../driver/display/display.h"
```

Include dependency graph for bullet.c:



Variables

- const Prototype Bullet PROGMEM

4.46.1 Variable Documentation

4.46.1.1 PROGMEM

```
const Prototype Bullet PROGMEM
```

Initial value:

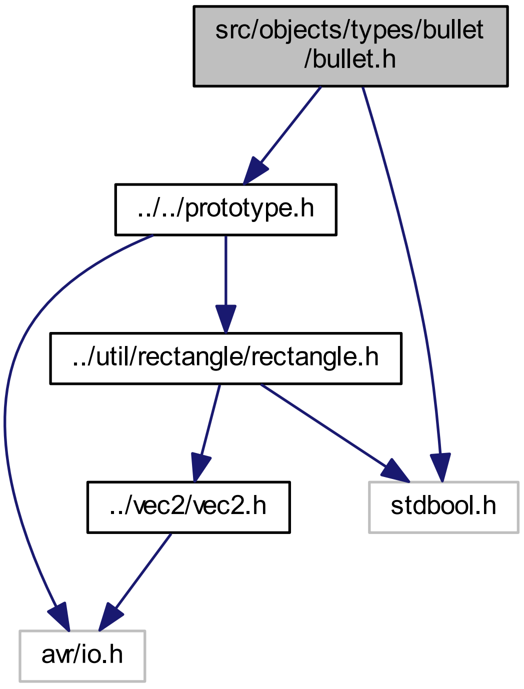
```
= {
    .tick = tick,
    .draw = draw,
    .size = BULLET_SIZE,
}
```

Definition at line 37 of file bullet.c.

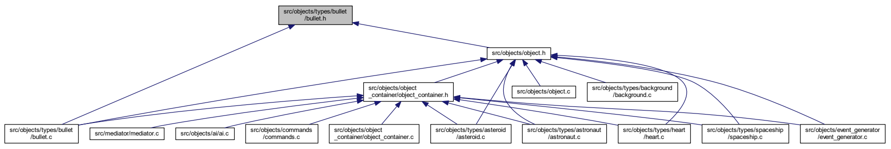
4.47 src/objects/types/bullet/bullet.h File Reference

```
#include "../../prototype.h"
#include <stdbool.h>
```

Include dependency graph for bullet.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct _bullet_t

Macros

- #define BULLET_SIZE ((Vec2){5, 1})

Variables

- const Prototype Bullet

4.47.1 Macro Definition Documentation

4.47.1.1 BULLET_SIZE

```
#define BULLET_SIZE ((Vec2){5, 1})
```

Definition at line 9 of file bullet.h.

4.47.2 Variable Documentation

4.47.2.1 Bullet

```
const Prototype Bullet
```

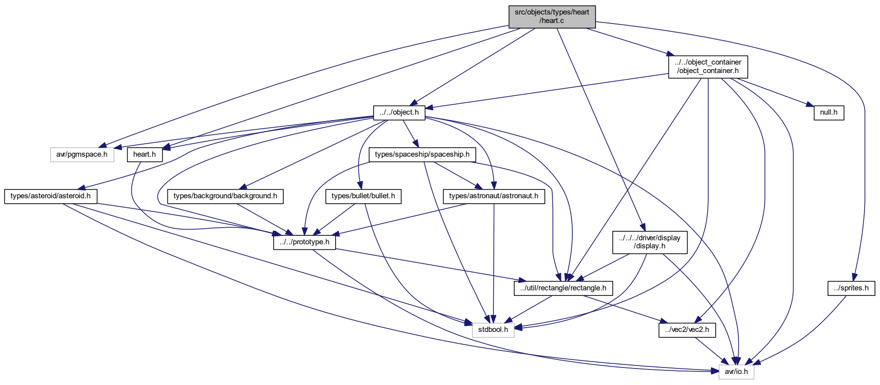
Definition at line 11 of file bullet.h.

Referenced by shootTurretOfSpaceship().

4.48 src/objects/types/heart/heart.c File Reference

```
#include "heart.h"
#include <avr/pgmspace.h>
#include "../../../driver/display/display.h"
#include "../sprites.h"
#include "../../object.h"
#include "../../object_container/object_container.h"
```

Include dependency graph for heart.c:


src/objects/types/heart/heart.c
../../object_container/object_container.h
../../object.h
null.h
avr/pgmspace.h
heart.h
types/spaceship/spaceship.h
types/asteroid/asteroid.h
types/background/background.h
types/bullet/bullet.h
types/astronaut/astronaut.h
../../../driver/display/display.h
../../prototype.h
../util/rectangle/rectangle.h
../sprites.h
stdbool.h
../vec2/vec2.h
avr/io.h

Variables

- const Prototype Heart PROGMEM

4.48.1 Variable Documentation

4.48.1.1 PROGMEM

```
const Prototype Heart PROGMEM
```

Initial value:

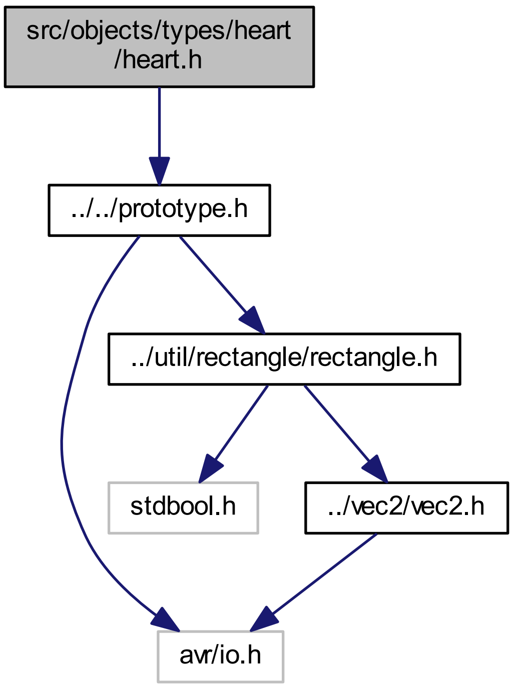
```
= {
    .tick = tick,
    .draw = draw,
    .size = HEART_SIZE,
}
```

Definition at line 29 of file heart.c.

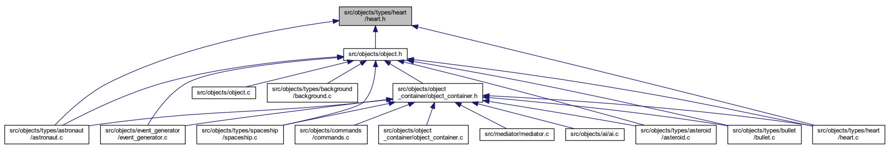
4.49 src/objects/types/heart/heart.h File Reference

`#include "../../prototype.h"`

Include dependency graph for heart.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct _heart_t

Macros

- #define HEART_SIZE ((Vec2){7, 6})
- #define TIME_TO_LIVE 128
- #define HEART_FRAME_COUNT 2
- #define HEART_ANIMATION_INTERVAL 12

Variables

- const Prototype Heart

4.49.1 Macro Definition Documentation

4.49.1.1 HEART_ANIMATION_INTERVAL

```
#define HEART_ANIMATION_INTERVAL 12
```

Definition at line 11 of file heart.h.

4.49.1.2 HEART_FRAME_COUNT

```
#define HEART_FRAME_COUNT 2
```

Definition at line 10 of file heart.h.

4.49.1.3 HEART_SIZE

```
#define HEART_SIZE ((Vec2){7, 6})
```

Definition at line 8 of file heart.h.

4.49.1.4 TIME_TO_LIVE

```
#define TIME_TO_LIVE 128
```

Definition at line 9 of file heart.h.

4.49.2 Variable Documentation

4.49.2.1 Heart

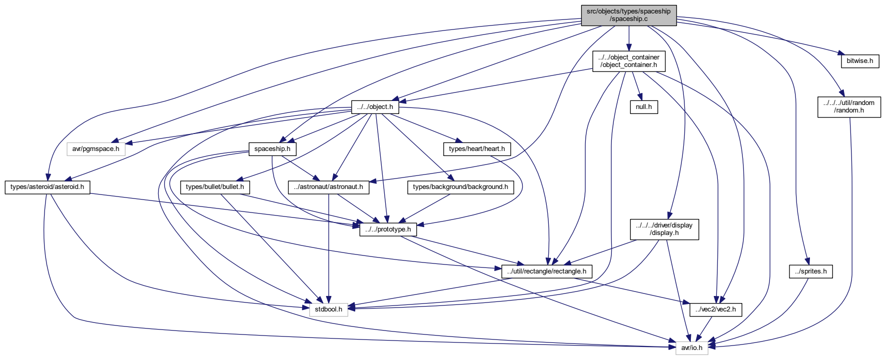
```
const Prototype Heart
```

Definition at line 13 of file heart.h.

4.50 src/objects/types/spaceship/spaceship.c File Reference

```
#include "spaceship.h"
#include <avr/pgmspace.h>
#include "../../object.h"
#include "../../../driver/display/display.h"
#include "../astronaut/astronaut.h"
#include "../asteroid/asteroid.h"
#include "../../object_container/object_container.h"
#include "../sprites.h"
#include "../../../util/vec2/vec2.h"
#include "../../../util/random/random.h"
#include "bitwise.h"
```

Include dependency graph for spaceship.c:


src/objects/types/spaceship/spaceship.c
../../object_container/object_container.h
bitwise.h
null.h
../../../util/random/random.h
../../object.h
avr/pgmspace.h
spaceship.h
types/heart/heart.h
types/asteroid/asteroid.h
types/bullet/bullet.h
../astronaut/astronaut.h
types/background/background.h
../../prototype.h
../../../driver/display/display.h
../util/rectangle/rectangle.h
../sprites.h
stdbool.h
../vec2/vec2.h
avr/io.h

Functions

- bool isOnUpperFloor (Rectangle boundingBox)
- bool isOnLowerFloor (Rectangle boundingBox)
- bool isBottomOnFloor (Rectangle boundingBox)
- bool isOnLadder (Rectangle boundingBox)
- bool isOnboard (Rectangle boundingBox)
- Rectangle getBoundingBoxOfSpaceshipPart (SpaceshipPart ∗part)
- void shootTurretOfSpaceship ()
- void onAsteroidMined ()
- void moveSpaceship (Vec2 direction)
- Action getPossibleActionFromSpaceship (Object ∗astronaut)
- void tick (Object ∗spaceship, uint8_t previousFrameTime)
- bool isSpaceshipPartActivated (SpaceshipPart ∗part)

Variables

- SpaceshipPart spaceshipParts [SPACESHIP_PART_COUNT]
- const Prototype Spaceship PROGMEM

4.50.1 Function Documentation

4.50.1.1 getBoundingBoxOfSpaceshipPart()

```
Rectangle getBoundingBoxOfSpaceshipPart (
              SpaceshipPart * part )
```

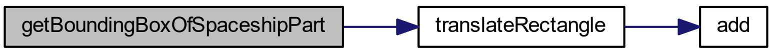
Definition at line 69 of file spaceship.c.

```
                                                               {
    return translateRectangle(part->boundingBox, spaceshipObject->position);
}
```

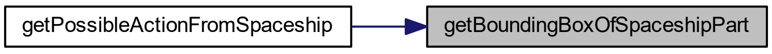
References SpaceshipPart::boundingBox, spaceshipObject, and translateRectangle().

Referenced by getPossibleActionFromSpaceship().

Here is the call graph for this function:



Here is the caller graph for this function:



4.50.1.2 getPossibleActionFromSpaceship()

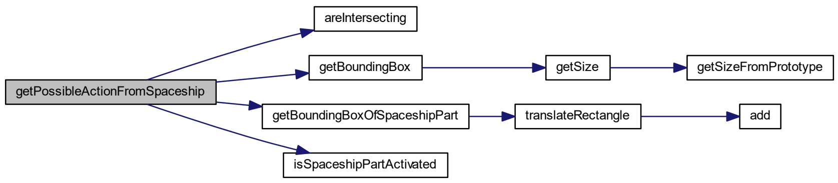
```
Action getPossibleActionFromSpaceship (
              Object * astronaut )
```

Definition at line 125 of file spaceship.c.

```
                                                         {
    for (uint8_t i = 0; i < SPACESHIP_PART_COUNT; i++) {
        SpaceshipPart* part = spaceshipParts + i;
        if (
            isSpaceshipPartActivated(part) && areIntersecting(getBoundingBoxOfSpaceshipPart(part),
    getBoundingBox(astronaut))) {
            return part->possibleAction;
        }
    }

    return noAction;
}
```

References areIntersecting(), getBoundingBox(), getBoundingBoxOfSpaceshipPart(), isSpaceshipPartActivated(), noAction, SpaceshipPart::possibleAction, SPACESHIP_PART_COUNT, and spaceshipParts.

Here is the call graph for this function:



4.50.1.3 isBottomOnFloor()

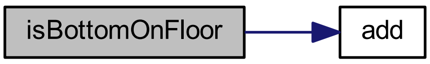
```
bool isBottomOnFloor (
            Rectangle boundingBox )
```

Definition at line 54 of file spaceship.c.

```
                                        {
    return (
        add(spaceshipObject->position, UPPER_FLOOR_BOUNDING_BOX.position).y +
    UPPER_FLOOR_BOUNDING_BOX.size.y == boundingBox.position.y + boundingBox.size.y ||
        add(spaceshipObject->position, LOWER_FLOOR_BOUNDING_BOX.position).y +
    LOWER_FLOOR_BOUNDING_BOX.size.y == boundingBox.position.y + boundingBox.size.y
    );
}
```

References add(), LOWER_FLOOR_BOUNDING_BOX, Rectangle::position, Rectangle::size, spaceshipObject, UPPER_FLOOR_BOUNDING_BOX, and Vec2::y.

Here is the call graph for this function:



4.50.1.4 isOnboard()

```
bool isOnboard (
            Rectangle boundingBox )
```

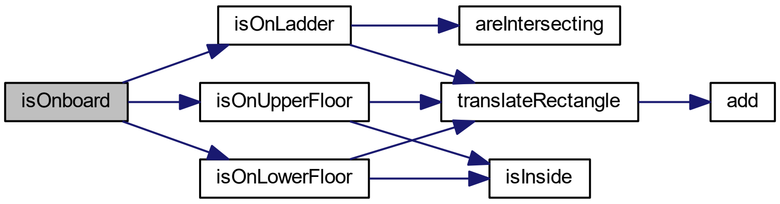
Definition at line 65 of file spaceship.c.

```
                                  {
    return isOnLowerFloor(boundingBox) || isOnUpperFloor(boundingBox) || isOnLadder(boundingBox);
}
```

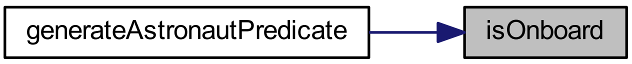
References isOnLadder(), isOnLowerFloor(), and isOnUpperFloor().

Referenced by generateAstronautPredicate().

Here is the call graph for this function:



Here is the caller graph for this function:



4.50.1.5 isOnLadder()

```
bool isOnLadder (
             Rectangle boundingBox )
```

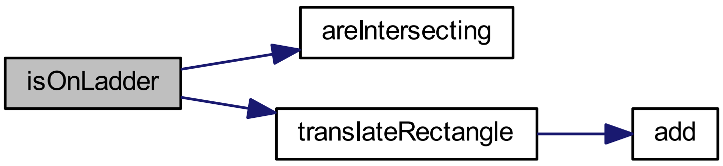
Definition at line 61 of file spaceship.c.

```
                                        {
    return areIntersecting(boundingBox, translateRectangle(LADDER_BOUNDING_BOX,
    spaceshipObject->position));
}
```

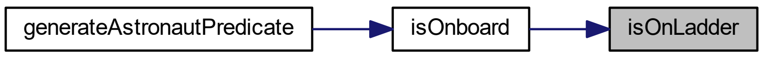
References areIntersecting(), LADDER_BOUNDING_BOX, spaceshipObject, and translateRectangle().

Referenced by isOnboard().

Here is the call graph for this function:



Here is the caller graph for this function:



4.50.1.6 isOnLowerFloor()

```
bool isOnLowerFloor (
             Rectangle boundingBox )
```

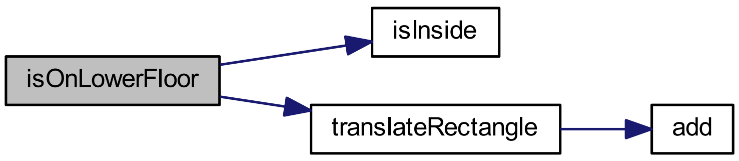
Definition at line 50 of file spaceship.c.

```
                                       {
      return isInside(boundingBox, translateRectangle(LOWER_FLOOR_BOUNDING_BOX,
      spaceshipObject->position));
 }
```

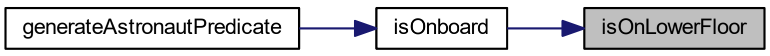
References isInside(), LOWER_FLOOR_BOUNDING_BOX, spaceshipObject, and translateRectangle().

Referenced by isOnboard().

Here is the call graph for this function:



Here is the caller graph for this function:



4.50.1.7 isOnUpperFloor()

```
bool isOnUpperFloor (
            Rectangle boundingBox )
```

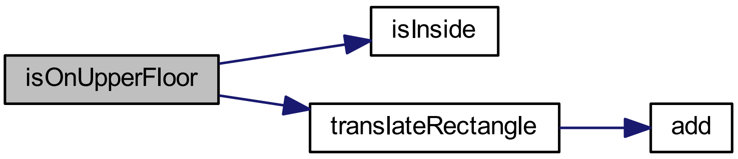
Definition at line 46 of file spaceship.c.

```
{
    return isInside(boundingBox, translateRectangle(UPPER_FLOOR_BOUNDING_BOX,
    spaceshipObject->position));
}
```

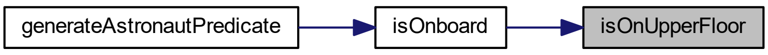
References isInside(), spaceshipObject, translateRectangle(), and UPPER_FLOOR_BOUNDING_BOX.

Referenced by isOnboard().

Here is the call graph for this function:



Here is the caller graph for this function:



4.50.1.8 isSpaceshipPartActivated()

```
bool isSpaceshipPartActivated (
            SpaceshipPart * part )
```

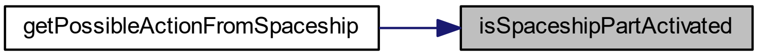
Definition at line 144 of file spaceship.c.

```
{
    return part->alwaysActiveDoNotDraw || ((spaceshipObject->as.spaceship.activatedParts » (part -
    spaceshipParts)) & 1);
}
```

References SpaceshipPart::alwaysActiveDoNotDraw, spaceshipObject, and spaceshipParts.

Referenced by getPossibleActionFromSpaceship().

Here is the caller graph for this function:



4.50.1.9 moveSpaceship()

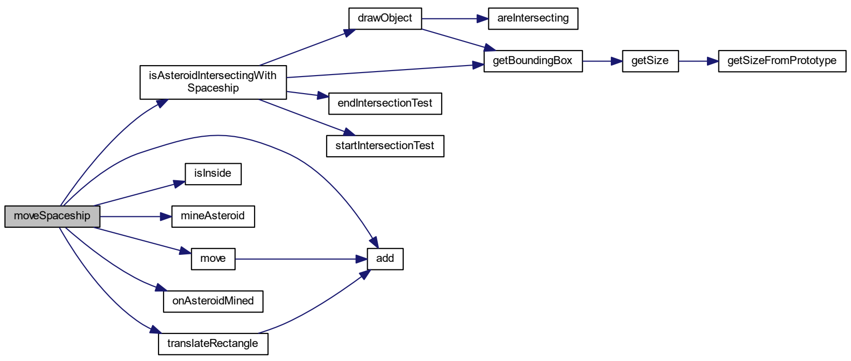
```
void moveSpaceship (
            Vec2 direction )
```

Definition at line 98 of file spaceship.c.

```
98                                     {
99      Vec2 proposedPosition = add(spaceshipObject->position, direction);
100
101      if (!isInside(translateRectangle(IN_VIEW_BOUNDING_BOX, proposedPosition), WINDOW)) {
102          return;
103      }
104
105      for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
106          if (objects[i].prototype == &Astronaut) {
107              move(objects + i, direction);
108          }
109      }
110
111      move(spaceshipObject, direction);
112
113      spaceshipObject->position = proposedPosition;
114
115      for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
116          if (objects[i].prototype == &Asteroid && isAsteroidIntersectingWithSpaceship(objects + i,
        spaceshipObject)) {
117              if (mineAsteroid(objects + i)) {
118                  spaceshipObject->as.spaceship.healthLoss += 2;
119                  onAsteroidMined();
120              }
121          }
122      }
123 }
```

References add(), Asteroid, Astronaut, IN_VIEW_BOUNDING_BOX, isAsteroidIntersectingWithSpaceship(), is↩ Inside(), mineAsteroid(), move(), OBJECT_COUNT, objects, onAsteroidMined(), spaceshipObject, translate↩ Rectangle(), and WINDOW.

Here is the call graph for this function:



4.50.1.10 onAsteroidMined()

```
void onAsteroidMined ( )
```

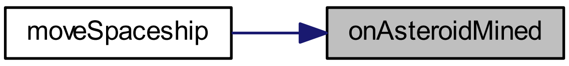
Definition at line 82 of file spaceship.c.

```
                     {
    switch (++spaceshipObject->as.spaceship.progress) {
        case hasBeds:
            setBit(spaceshipObject->as.spaceship.activatedParts, BEDS_INDEX);
            break;
        case hasTurret:
            setBit(spaceshipObject->as.spaceship.activatedParts, TURRET_CONTROLLER_INDEX);
            break;
        case hasTable:
            setBit(spaceshipObject->as.spaceship.activatedParts, TABLE_INDEX);
            break;
        default:
            break;
    }
}
```

References BEDS_INDEX, hasBeds, hasTable, hasTurret, setBit, spaceshipObject, TABLE_INDEX, and TURRE↩ T_CONTROLLER_INDEX.

Referenced by moveSpaceship().

Here is the caller graph for this function:



4.50.1.11 shootTurretOfSpaceship()

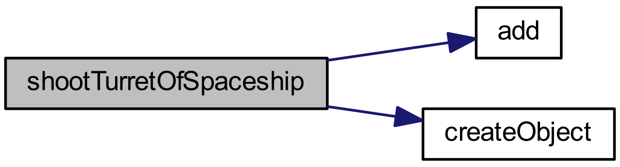
```
void shootTurretOfSpaceship ( )
```

Definition at line 73 of file spaceship.c.

```
                            {
    Object* bullet = getEmptyObjectSpace();
    if (getEmptyObjectSpace() != NULL && spaceshipObject->as.spaceship.healthLoss < MAX_HEALTH - 1) {
        createObject(&Bullet, bullet);
        bullet->position = add(TURRET_POSITION, spaceshipObject->position);
        spaceshipObject->as.spaceship.healthLoss++;
    }
}
```

References add(), Bullet, createObject(), getEmptyObjectSpace, MAX_HEALTH, NULL, _object_t::position, spaceshipObject, and TURRET_POSITION.

Here is the call graph for this function:



4.50.1.12 tick()

```
void tick (
            Object * spaceship,
            uint8_t previousFrameTime )
```

Definition at line 137 of file spaceship.c.

```
                                                          {
    flickerState = !flickerState;
    if (spaceship->as.spaceship.healthLoss >= MAX_HEALTH) {
        spaceship->as.spaceship.healthLoss++;
    }
}
```

4.50.2 Variable Documentation

4.50.2.1 PROGMEM

```
const Prototype Spaceship PROGMEM
```

Initial value:

```
= {
    .tick = tick,
    .draw = draw,
    .size = SPACESHIP_SIZE,
}
```

Definition at line 204 of file spaceship.c.

4.50.2.2 spaceshipParts

```
SpaceshipPart spaceshipParts[SPACESHIP_PART_COUNT]
```

Initial value:

```
= {
    [TABLE_INDEX] = (SpaceshipPart) {
        {{7, 8}, {3, 3}},
        table[0],
        showLove,
        false
    },
    [BEDS_INDEX] = (SpaceshipPart) {
        {{3, 12}, {8, 6}},
        beds[0],
        repairingSpaceship,
        false
    },
    [COMMAND_PANEL_INDEX] = (SpaceshipPart) {
        {{26, 7}, {7, 4}},
        NULL,
        controllingSpaceship,
        true,
    },
    [TURRET_CONTROLLER_INDEX] = (SpaceshipPart) {
        {{26, 12}, {7, 6}},
        turret_controller[0],
        shootTurret,
        false
    }
}
```

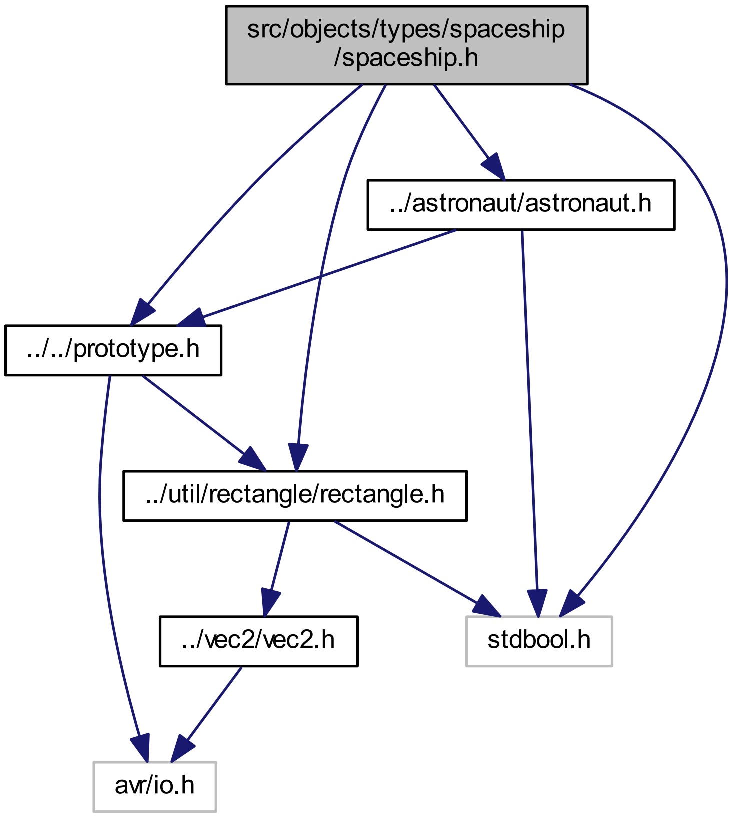
Definition at line 18 of file spaceship.c.

Referenced by getPossibleActionFromSpaceship(), and isSpaceshipPartActivated().

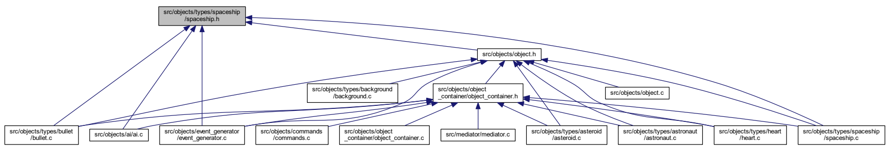
4.51 src/objects/types/spaceship/spaceship.h File Reference

```
#include "../../prototype.h"
#include "../../../util/rectangle/rectangle.h"
#include "../astronaut/astronaut.h"
#include <stdbool.h>
```

Include dependency graph for spaceship.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct SpaceshipPart
- struct _spaceship_t

Macros

- #define SPACESHIP_SIZE ((Vec2){36, 23})
- #define IN_VIEW_BOUNDING_BOX ((Rectangle){(Vec2){7, 4}, (Vec2){22, 15}})

- #define UPPER_FLOOR_BOUNDING_BOX ((Rectangle){(Vec2){8, 5}, (Vec2){19, 6}})
- #define LOWER_FLOOR_BOUNDING_BOX ((Rectangle){(Vec2){5, 12}, (Vec2){23, 6}})
- #define EXHAUST_BOUNDING_BOX ((Rectangle){(Vec2){-4, 9}, (Vec2){5, 5}})
- #define TURRET_POSITION ((Vec2){35, 11})
- #define LADDER_BOUNDING_BOX ((Rectangle){(Vec2){12, 10}, (Vec2){1, 4}})
- #define BOBBING_INTERVAL 130
- #define SPACESHIP_PART_COUNT 4
- #define TABLE_INDEX 3
- #define BEDS_INDEX 2
- #define COMMAND_PANEL_INDEX 1
- #define TURRET_CONTROLLER_INDEX 0
- #define BAR_END_POSITION ((Vec2){33, 11})
- #define BAR_LENGTH 4
- #define MAX_HEALTH 8

Enumerations

- enum Progress { hasBeds = 1, hasTurret = 3, hasTable = 5, hasFullCrew = 9 }

Functions

- bool isOnboard (Rectangle boundingBox)
- void moveSpaceship (Vec2 direction)
- Rectangle getBoundingBoxOfSpaceshipPart (SpaceshipPart ∗part)
- bool isBottomOnFloor (Rectangle boundingBox)
- bool isOnUpperFloor (Rectangle boundingBox)
- bool isOnLowerFloor (Rectangle boundingBox)
- bool isOnLadder (Rectangle boundingBox)
- void onAsteroidMined ()
- bool isSpaceshipPartActivated (SpaceshipPart ∗part)
- void shootTurretOfSpaceship ()
- Action getPossibleActionFromSpaceship (Object ∗astronaut)

Variables

- SpaceshipPart spaceshipParts [SPACESHIP_PART_COUNT]
- const Prototype Spaceship

4.51.1 Macro Definition Documentation

4.51.1.1 BAR_END_POSITION

```
#define BAR_END_POSITION ((Vec2){33, 11})
```

Definition at line 31 of file spaceship.h.

4.51.1.2 BAR_LENGTH

```
#define BAR_LENGTH 4
```

Definition at line 32 of file spaceship.h.

4.51.1.3 BEDS_INDEX

```
#define BEDS_INDEX 2
```

Definition at line 27 of file spaceship.h.

4.51.1.4 BOBBING_INTERVAL

```
#define BOBBING_INTERVAL 130
```

Definition at line 23 of file spaceship.h.

4.51.1.5 COMMAND_PANEL_INDEX

```
#define COMMAND_PANEL_INDEX 1
```

Definition at line 28 of file spaceship.h.

4.51.1.6 EXHAUST_BOUNDING_BOX

```
#define EXHAUST_BOUNDING_BOX ((Rectangle){(Vec2){-4, 9}, (Vec2){5, 5}})
```

Definition at line 17 of file spaceship.h.

4.51.1.7 IN_VIEW_BOUNDING_BOX

```
#define IN_VIEW_BOUNDING_BOX ((Rectangle){(Vec2){7, 4}, (Vec2){22, 15}})
```

Definition at line 14 of file spaceship.h.

4.51.1.8 LADDER_BOUNDING_BOX

```
#define LADDER_BOUNDING_BOX ((Rectangle){(Vec2){12, 10}, (Vec2){1, 4}})
```

Definition at line 21 of file spaceship.h.

4.51.1.9 LOWER_FLOOR_BOUNDING_BOX

```
#define LOWER_FLOOR_BOUNDING_BOX ((Rectangle){(Vec2){5, 12}, (Vec2){23, 6}})
```

Definition at line 16 of file spaceship.h.

4.51.1.10 MAX_HEALTH

```
#define MAX_HEALTH 8
```

Definition at line 33 of file spaceship.h.

4.51.1.11 SPACESHIP_PART_COUNT

```
#define SPACESHIP_PART_COUNT 4
```

Definition at line 24 of file spaceship.h.

4.51.1.12 SPACESHIP_SIZE

```
#define SPACESHIP_SIZE ((Vec2){36, 23})
```

Definition at line 12 of file spaceship.h.

4.51.1.13 TABLE_INDEX

```
#define TABLE_INDEX 3
```

Definition at line 26 of file spaceship.h.

4.51.1.14 TURRET_CONTROLLER_INDEX

```
#define TURRET_CONTROLLER_INDEX 0
```

Definition at line 29 of file spaceship.h.

4.51.1.15 TURRET_POSITION

```
#define TURRET_POSITION ((Vec2){35, 11})
```

Definition at line 19 of file spaceship.h.

4.51.1.16 UPPER_FLOOR_BOUNDING_BOX

```
#define UPPER_FLOOR_BOUNDING_BOX ((Rectangle){(Vec2){8, 5}, (Vec2){19, 6}})
```

Definition at line 15 of file spaceship.h.

4.51.2 Enumeration Type Documentation

4.51.2.1 Progress

```
enum Progress
```

Enumerator

hasBeds	
hasTurret	
hasTable	
hasFullCrew	

Definition at line 45 of file spaceship.h.

```
             {
    hasBeds = 1,
    hasTurret = 3,
    hasTable = 5,
    hasFullCrew = 9
} Progress;
```

4.51.3 Function Documentation

4.51.3.1 getBoundingBoxOfSpaceshipPart()

```
Rectangle getBoundingBoxOfSpaceshipPart (
            SpaceshipPart * part )
```

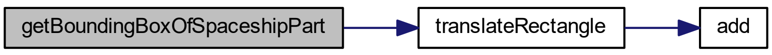
Definition at line 69 of file spaceship.c.

```
                                                    {
    return translateRectangle(part->boundingBox, spaceshipObject->position);
}
```

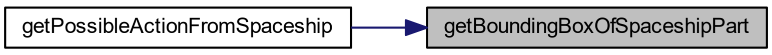
References SpaceshipPart::boundingBox, spaceshipObject, and translateRectangle().

Referenced by getPossibleActionFromSpaceship().

Here is the call graph for this function:



Here is the caller graph for this function:



4.51.3.2 getPossibleActionFromSpaceship()

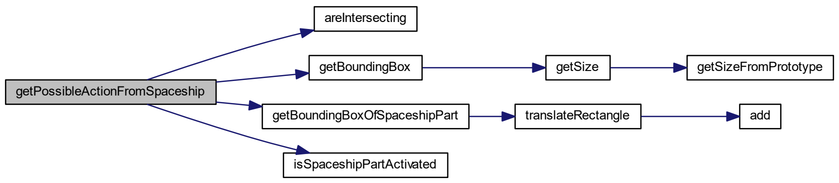
```
Action getPossibleActionFromSpaceship (
            Object * astronaut )
```

Definition at line 125 of file spaceship.c.

```
                                                {
    for (uint8_t i = 0; i < SPACESHIP_PART_COUNT; i++) {
        SpaceshipPart* part = spaceshipParts + i;
        if (
            isSpaceshipPartActivated(part) && areIntersecting(getBoundingBoxOfSpaceshipPart(part),
    getBoundingBox(astronaut))) {
            return part->possibleAction;
        }
    }

    return noAction;
}
```

References areIntersecting(), getBoundingBox(), getBoundingBoxOfSpaceshipPart(), isSpaceshipPartActivated(), noAction, SpaceshipPart::possibleAction, SPACESHIP_PART_COUNT, and spaceshipParts.

Here is the call graph for this function:



4.51.3.3 isBottomOnFloor()

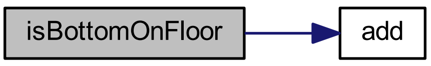
```
bool isBottomOnFloor (
              Rectangle boundingBox )
```

Definition at line 54 of file spaceship.c.

```
                                            {
    return (
        add(spaceshipObject->position, UPPER_FLOOR_BOUNDING_BOX.position).y +
    UPPER_FLOOR_BOUNDING_BOX.size.y == boundingBox.position.y + boundingBox.size.y ||
        add(spaceshipObject->position, LOWER_FLOOR_BOUNDING_BOX.position).y +
    LOWER_FLOOR_BOUNDING_BOX.size.y == boundingBox.position.y + boundingBox.size.y
    );
}
```

References add(), LOWER_FLOOR_BOUNDING_BOX, Rectangle::position, Rectangle::size, spaceshipObject, UPPER_FLOOR_BOUNDING_BOX, and Vec2::y.

Here is the call graph for this function:



4.51.3.4 isOnboard()

```
bool isOnboard (
              Rectangle boundingBox )
```

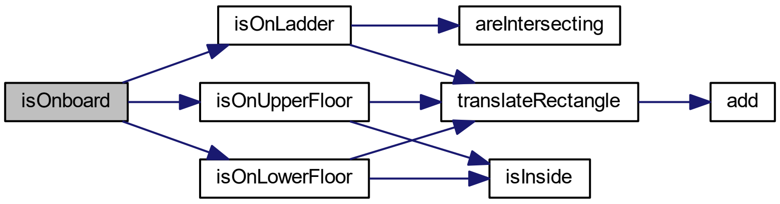
Definition at line 65 of file spaceship.c.

```
                                      {
    return isOnLowerFloor(boundingBox) || isOnUpperFloor(boundingBox) || isOnLadder(boundingBox);
}
```

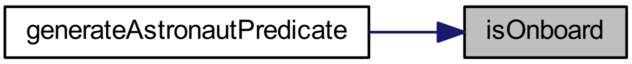
References isOnLadder(), isOnLowerFloor(), and isOnUpperFloor().

Referenced by generateAstronautPredicate().

Here is the call graph for this function:



Here is the caller graph for this function:



4.51.3.5 isOnLadder()

```
bool isOnLadder (
             Rectangle boundingBox )
```

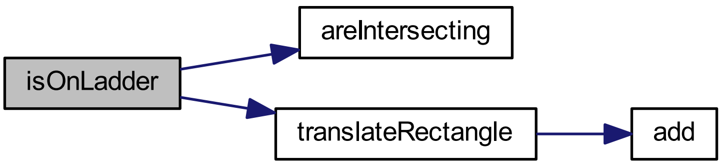
Definition at line 61 of file spaceship.c.

```
                                         {
    return areIntersecting(boundingBox, translateRectangle(LADDER_BOUNDING_BOX,
    spaceshipObject->position));
}
```

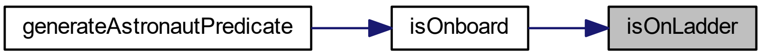
References areIntersecting(), LADDER_BOUNDING_BOX, spaceshipObject, and translateRectangle().

Referenced by isOnboard().

Here is the call graph for this function:



Here is the caller graph for this function:



4.51.3.6 isOnLowerFloor()

```
bool isOnLowerFloor (
            Rectangle boundingBox )
```

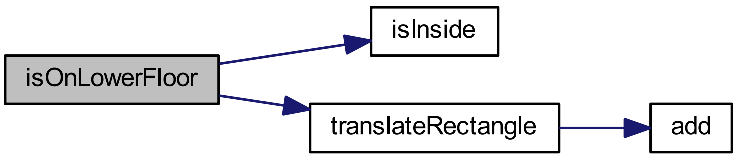
Definition at line 50 of file spaceship.c.

```
50                                              {
51     return isInside(boundingBox, translateRectangle(LOWER_FLOOR_BOUNDING_BOX,
       spaceshipObject->position));
52 }
```

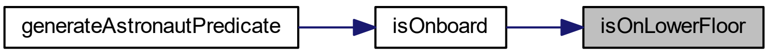
References isInside(), LOWER_FLOOR_BOUNDING_BOX, spaceshipObject, and translateRectangle().

Referenced by isOnboard().

Here is the call graph for this function:



Here is the caller graph for this function:



4.51.3.7 isOnUpperFloor()

```
bool isOnUpperFloor (
            Rectangle boundingBox )
```

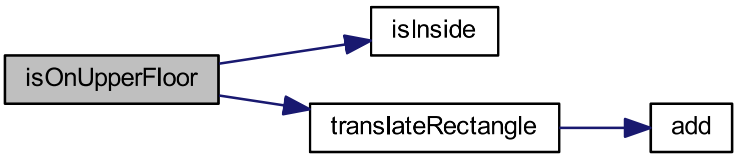
Definition at line 46 of file spaceship.c.

```
                                        {
    return isInside(boundingBox, translateRectangle(UPPER_FLOOR_BOUNDING_BOX,
    spaceshipObject->position));
}
```

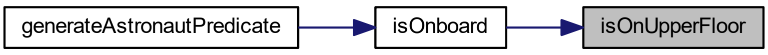
References isInside(), spaceshipObject, translateRectangle(), and UPPER_FLOOR_BOUNDING_BOX.

Referenced by isOnboard().

Here is the call graph for this function:



Here is the caller graph for this function:



4.51.3.8 isSpaceshipPartActivated()

```
bool isSpaceshipPartActivated (
            SpaceshipPart * part )
```

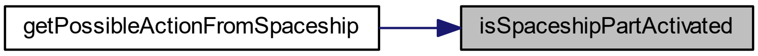
Definition at line 144 of file spaceship.c.

```
                                            {
     return part->alwaysActiveDoNotDraw || ((spaceshipObject->as.spaceship.activatedParts » (part -
    spaceshipParts)) & 1);
}
```

References SpaceshipPart::alwaysActiveDoNotDraw, spaceshipObject, and spaceshipParts.

Referenced by getPossibleActionFromSpaceship().

Here is the caller graph for this function:



4.51.3.9 moveSpaceship()

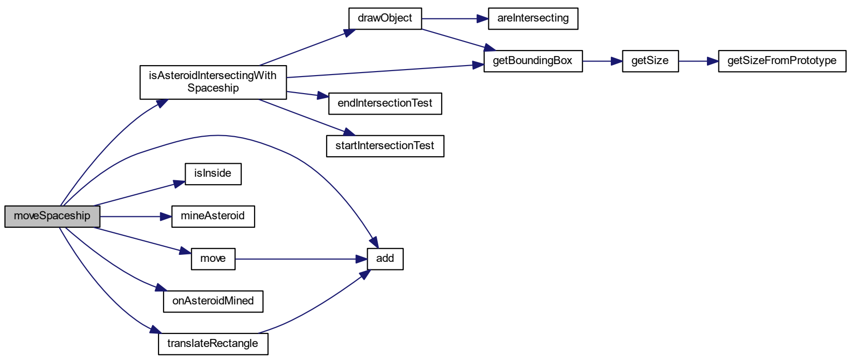
```
void moveSpaceship (
            Vec2 direction )
```

Definition at line 98 of file spaceship.c.

```
                                    {
    Vec2 proposedPosition = add(spaceshipObject->position, direction);

     if (!isInside(translateRectangle(IN_VIEW_BOUNDING_BOX, proposedPosition), WINDOW)) {
         return;
     }

     for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
         if (objects[i].prototype == &Astronaut) {
             move(objects + i, direction);
         }
     }

     move(spaceshipObject, direction);

     spaceshipObject->position = proposedPosition;

     for (uint8_t i = 0; i < OBJECT_COUNT; i++) {
         if (objects[i].prototype == &Asteroid && isAsteroidIntersectingWithSpaceship(objects + i,
    spaceshipObject)) {
             if (mineAsteroid(objects + i)) {
                 spaceshipObject->as.spaceship.healthLoss += 2;
                 onAsteroidMined();
             }
         }
     }
}
```

References add(), Asteroid, Astronaut, IN_VIEW_BOUNDING_BOX, isAsteroidIntersectingWithSpaceship(), is←
Inside(), mineAsteroid(), move(), OBJECT_COUNT, objects, onAsteroidMined(), spaceshipObject, translate←
Rectangle(), and WINDOW.

Here is the call graph for this function:



4.51.3.10 onAsteroidMined()

```
void onAsteroidMined ( )
```

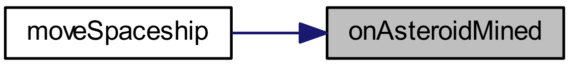
Definition at line 82 of file spaceship.c.

```
                          {
    switch (++spaceshipObject->as.spaceship.progress) {
        case hasBeds:
            setBit(spaceshipObject->as.spaceship.activatedParts, BEDS_INDEX);
            break;
        case hasTurret:
            setBit(spaceshipObject->as.spaceship.activatedParts, TURRET_CONTROLLER_INDEX);
            break;
        case hasTable:
            setBit(spaceshipObject->as.spaceship.activatedParts, TABLE_INDEX);
            break;
        default:
            break;
    }
}
```

References BEDS_INDEX, hasBeds, hasTable, hasTurret, setBit, spaceshipObject, TABLE_INDEX, and TURRE↩T_CONTROLLER_INDEX.

Referenced by moveSpaceship().

Here is the caller graph for this function:



4.51.3.11 shootTurretOfSpaceship()

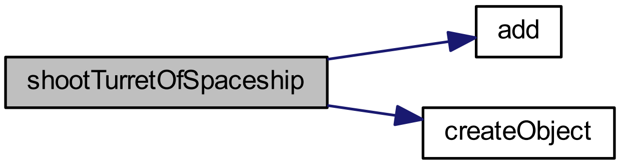
```
void shootTurretOfSpaceship ( )
```

Definition at line 73 of file spaceship.c.

```
                                 {
    Object* bullet = getEmptyObjectSpace();
    if (getEmptyObjectSpace() != NULL && spaceshipObject->as.spaceship.healthLoss < MAX_HEALTH - 1) {
        createObject(&Bullet, bullet);
        bullet->position = add(TURRET_POSITION, spaceshipObject->position);
        spaceshipObject->as.spaceship.healthLoss++;
    }
}
```

References add(), Bullet, createObject(), getEmptyObjectSpace, MAX_HEALTH, NULL, _object_t::position, spaceshipObject, and TURRET_POSITION.

Here is the call graph for this function:



4.51.4 Variable Documentation

4.51.4.1 Spaceship

```
const Prototype Spaceship
```

Definition at line 43 of file spaceship.h.

Referenced by createObjects(), and generateAsteroidPredicate().

4.51.4.2 spaceshipParts

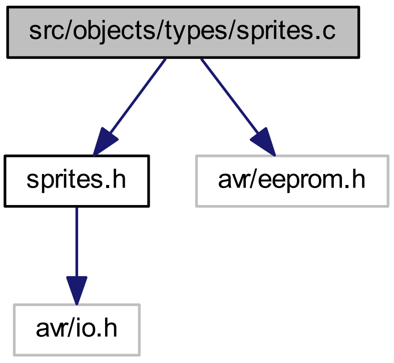
```
SpaceshipPart spaceshipParts[SPACESHIP_PART_COUNT]
```

Definition at line 41 of file spaceship.h.

4.52 src/objects/types/sprites.c File Reference

```
#include "sprites.h"
#include <avr/eeprom.h>
```

Include dependency graph for sprites.c:



Variables

- const uint16_t small_character_moving[4][5][1] EEMEM = {{{0x606},{0x1f1d},{0xf0b},{0x1f1d},{0x606}},{{0x606},{0x1f1d},{0xf0b

  AUTO-GENERATED.

4.52.1 Variable Documentation

4.52.1.1 EEMEM

```
const uint16_t turret_controller [1][7][1] EEMEM = {{{0x606},{0x1f1d},{0xf0b},{0x1f1d},{0x606}},{{0x606},{0x1f
```

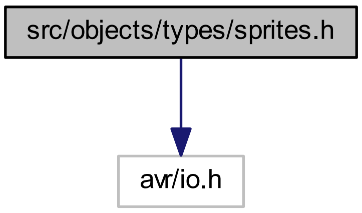
AUTO-GENERATED.

Definition at line 8 of file sprites.c.

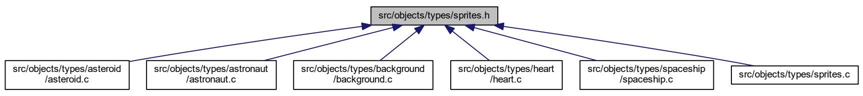
4.53 src/objects/types/sprites.h File Reference

```
#include <avr/io.h>
```

Include dependency graph for sprites.h:



This graph shows which files directly or indirectly include this file:


src/objects/types/sprites.h
src/objects/types/asteroid/asteroid.c
src/objects/types/astronaut/astronaut.c
src/objects/types/background/background.c
src/objects/types/heart/heart.c
src/objects/types/spaceship/spaceship.c
src/objects/types/sprites.c

Variables

- const uint16_t small_character_moving [4][5][1]

  AUTO-GENERATED.
- const uint16_t spaceship_idle [1][36][3] = {{{0x404,0x1414,0x1010},{0x404,0x3e3e,0x1010},{0xceca,0xffff,0x3828},{0xee6a,0xff
- const uint16_t table [1][3][1] = {{{0x707},{0x101},{0x101}}}
- const uint16_t exhaust [1][5][1] = {{{0x404},{0xe0e},{0xe0e},{0x1f1f},{0x404}}}
- const uint16_t small_asteroid [7][8][1] = {{{0x1c1c},{0x7e7e},{0xfef2},{0xfffb},{0xffff},{0xffdf},{0x7e7e},{0x3c3c}},{{0x0},{0x3c3c},
- const uint16_t stars [3][3][1] = {{{0x202},{0x707},{0x202}},{{0x505},{0x202},{0x505}},{{0x0},{0x303},{0x303}}}
- const uint16_t heart_blinking [2][7][1] = {{{0x606},{0x909},{0x1111},{0x2222},{0x1111},{0x909},{0x606}},{{0x606},{0xf0f},{0x1f1f
- const uint16_t beds [1][8][1] = {{{0x707},{0xc0c},{0x2424},{0x2424},{0x2424},{0x2424},{0x2424},{0x2424}}}
- const uint16_t turret_controller [1][7][1] = {{{0x3030},{0x808},{0xf0f},{0x101},{0x101},{0x101},{0x101}}}

4.53.1 Variable Documentation

4.53.1.1 beds

```
const uint16_t beds = {{{0x707},{0xc0c},{0x2424},{0x2424},{0x2424},{0x2424},{0x2424},{0x2424}}}
```

Definition at line 16 of file sprites.h.

4.53.1.2 exhaust

```
const uint16_t exhaust = {{{0x404},{0xe0e},{0xe0e},{0x1f1f},{0x404}}}
```

Definition at line 12 of file sprites.h.

4.53.1.3 heart_blinking

```
const uint16_t heart_blinking = {{{0x606},{0x909},{0x1111},{0x2222},{0x1111},{0x909},{0x606}},{{0x606},{0xf0f}
```

Definition at line 15 of file sprites.h.

4.53.1.4 small_asteroid

```
const uint16_t small_asteroid = {{{0x1c1c},{0x7e7e},{0xfef2},{0xfffb},{0xffff},{0xffdf},{0x7e7e},{0x3c3c}},{{0
```

Definition at line 13 of file sprites.h.

4.53.1.5 small_character_moving

```
const uint16_t small_character_moving[4][5][1]
```

AUTO-GENERATED.

Definition at line 9 of file sprites.h.

4.53.1.6 spaceship_idle

```
const uint16_t spaceship_idle = {{{0x404,0x1414,0x1010},{0x404,0x3e3e,0x1010},{0xceca,0xffff,0x3828},{0xee6a,0
```

Definition at line 10 of file sprites.h.

4.53.1.7 stars

```
const uint16_t stars = {{{0x202},{0x707},{0x202}},{{0x505},{0x202},{0x505}},{{0x0},{0x303},{0x303}}}
```

Definition at line 14 of file sprites.h.

4.53.1.8 table

```
const uint16_t table = {{{0x707},{0x101},{0x101}}}
```

Definition at line 11 of file sprites.h.

4.53.1.9 turret_controller

```
const uint16_t turret_controller = {{{0x3030},{0x808},{0xf0f},{0x101},{0x101},{0x101},{0x101}}}
```

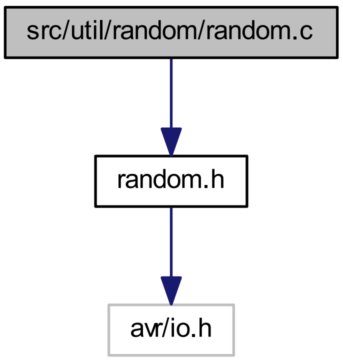
Definition at line 17 of file sprites.h.

4.54 src/Project.txt File Reference

4.55 src/util/random/random.c File Reference

```
#include "random.h"
```

Include dependency graph for random.c:



Functions

- uint8_t getRandomNumber ()

4.55.1 Function Documentation

4.55.1.1 getRandomNumber()

```
uint8_t getRandomNumber ( )
```

Simple LFSR with some improvements to enhance distribution while maintaining short execution time

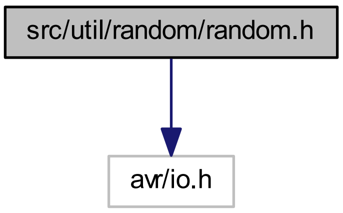
Definition at line 12 of file random.c.

```
12                             {
13     return (uint8_t)(getRandom16bitNumberModuloPrime() ^ getRandom16bitNumberModuloPrime());
14 }
```

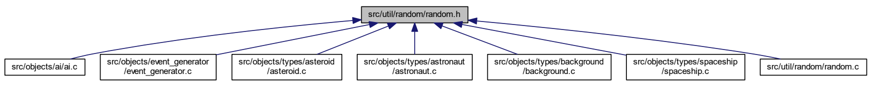
4.56 src/util/random/random.h File Reference

```
#include <avr/io.h>
```

Include dependency graph for random.h:



This graph shows which files directly or indirectly include this file:


src/util/random/random.h
src/objects/ai/ai.c
src/objects/event_generator /event_generator.c
src/objects/types/asteroid /asteroid.c
src/objects/types/astronaut /astronaut.c
src/objects/types/background /background.c
src/objects/types/spaceship /spaceship.c
src/util/random/random.c

Macros

- #define SEED 42
  Mustn't be zero, should be lower than 65535.

Functions

- uint8_t getRandomNumber ()

4.56.1 Macro Definition Documentation

4.56.1.1 SEED

```
#define SEED 42
```

Mustn't be zero, should be lower than 65535.

Definition at line 9 of file random.h.

4.56.2 Function Documentation

4.56.2.1 getRandomNumber()

```
uint8_t getRandomNumber ( )
```

Simple LFSR with some improvements to enhance distribution while maintaining short execution time

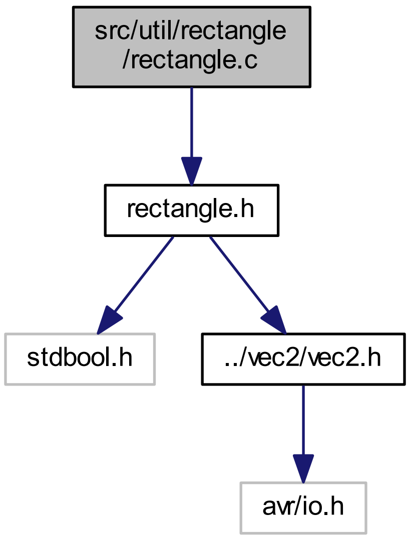
Definition at line 12 of file random.c.

```
                       {
    return (uint8_t)(getRandom16bitNumberModuloPrime() ^ getRandom16bitNumberModuloPrime());
}
```

4.57 src/util/rectangle/rectangle.c File Reference

```
#include "rectangle.h"
```

Include dependency graph for rectangle.c:



Functions

- bool areIntersecting (Rectangle r1, Rectangle r2)
- bool isInside (Rectangle inner, Rectangle outer)

  Return true only if inner is fully inside of outer.
- Vec2 getCenter (Rectangle r)

  Return the geometrical middle point of the given rectangle.
- Rectangle translateRectangle (Rectangle r, Vec2 translate)

  Return a new rectangle shifted by vector translate.

4.57.1 Function Documentation

4.57.1.1 areIntersecting()

```
bool areIntersecting (
            Rectangle r1,
            Rectangle r2 )
```

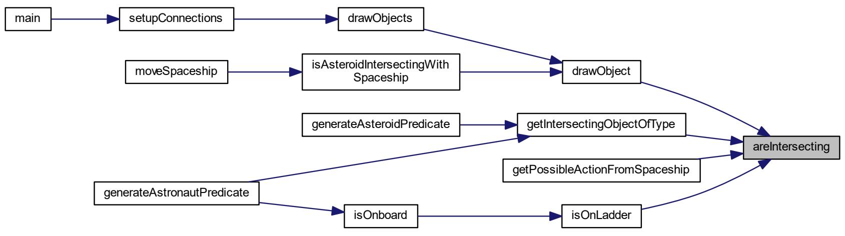
Definition at line 4 of file rectangle.c.

```
                                                        {
    return (
        r1.position.x < r2.position.x + r2.size.x &&
        r1.position.x + r1.size.x > r2.position.x &&
        r1.position.y < r2.position.y + r2.size.y &&
        r1.position.y + r1.size.y > r2.position.y
    );
}
```

References Rectangle::position, Rectangle::size, Vec2::x, and Vec2::y.

Referenced by drawObject(), getIntersectingObjectOfType(), getPossibleActionFromSpaceship(), and isOnLadder().

Here is the caller graph for this function:


main
setupConnections
drawObjects
moveSpaceship
isAsteroidIntersectingWith Spaceship
drawObject
generateAsteroidPredicate
getIntersectingObjectOfType
areIntersecting
getPossibleActionFromSpaceship
generateAstronautPredicate
isOnboard
isOnLadder

4.57.1.2 getCenter()

```
Vec2 getCenter (
            Rectangle r )
```

Return the geometrical middle point of the given rectangle.

Definition at line 22 of file rectangle.c.

```
                                {
    return (Vec2) {
        r.position.x + r.size.x / 2,
        r.position.y + r.size.y / 2
    };
}
```

References Rectangle::position, Rectangle::size, Vec2::x, and Vec2::y.

4.57.1.3 isInside()

```
bool isInside (
            Rectangle inner,
            Rectangle outer )
```

Return true only if inner is fully inside of outer.

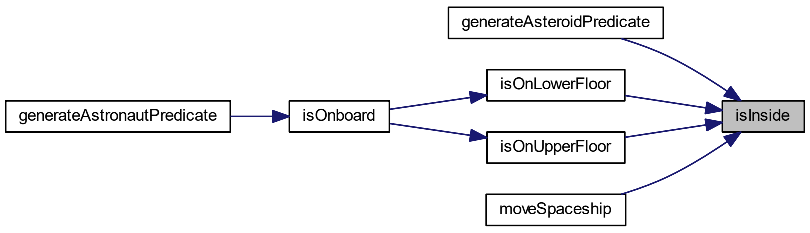
Definition at line 13 of file rectangle.c.

```
                                                     {
    return (
        outer.position.x <= inner.position.x &&
        inner.position.x + inner.size.x <= outer.position.x + outer.size.x &&
        outer.position.y <= inner.position.y &&
        inner.position.y + inner.size.y <= outer.position.y + outer.size.y
    );
}
```

References Rectangle::position, Rectangle::size, Vec2::x, and Vec2::y.

Referenced by generateAsteroidPredicate(), isOnLowerFloor(), isOnUpperFloor(), and moveSpaceship().

Here is the caller graph for this function:



4.57.1.4 translateRectangle()

```
Rectangle translateRectangle (
            Rectangle r,
            Vec2 translate )
```

Return a new rectangle shifted by vector translate.

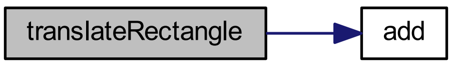
Definition at line 29 of file rectangle.c.

```
                                                      {
    return (Rectangle) {
        add(r.position, translate),
        r.size
    };
}
```

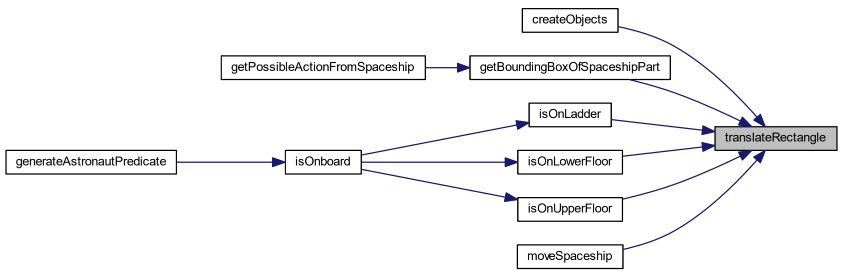
References add(), Rectangle::position, and Rectangle::size.

Referenced by createObjects(), getBoundingBoxOfSpaceshipPart(), isOnLadder(), isOnLowerFloor(), isOnUpper←Floor(), and moveSpaceship().

Here is the call graph for this function:



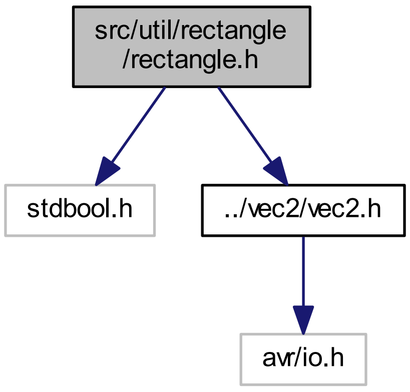
Here is the caller graph for this function:



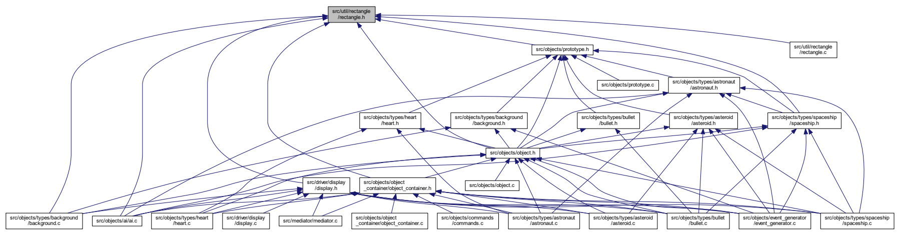
4.58 src/util/rectangle/rectangle.h File Reference

```
#include <stdbool.h>
#include "../vec2/vec2.h"
```

Include dependency graph for rectangle.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct Rectangle

Functions

- bool areIntersecting (Rectangle r1, Rectangle r2)
- bool isInside (Rectangle inner, Rectangle outer)

  Return true only if inner is fully inside of outer.
- Vec2 getCenter (Rectangle r)

  Return the geometrical middle point of the given rectangle.
- Rectangle translateRectangle (Rectangle r, Vec2 translate)

  Return a new rectangle shifted by vector translate.

4.58.1 Function Documentation

4.58.1.1 areIntersecting()

```
bool areIntersecting (
            Rectangle r1,
            Rectangle r2 )
```

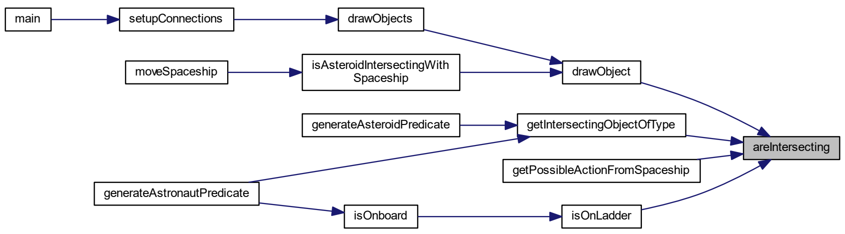
Definition at line 4 of file rectangle.c.

```
4                                                    {
5     return (
6         r1.position.x < r2.position.x + r2.size.x &&
7         r1.position.x + r1.size.x > r2.position.x &&
8         r1.position.y < r2.position.y + r2.size.y &&
9         r1.position.y + r1.size.y > r2.position.y
10     );
11 }
```

References Rectangle::position, Rectangle::size, Vec2::x, and Vec2::y.

Referenced by drawObject(), getIntersectingObjectOfType(), getPossibleActionFromSpaceship(), and isOnLadder().

Here is the caller graph for this function:


main
setupConnections
drawObjects
moveSpaceship
isAsteroidIntersectingWith Spaceship
drawObject
generateAsteroidPredicate
getIntersectingObjectOfType
areIntersecting
getPossibleActionFromSpaceship
generateAstronautPredicate
isOnboard
isOnLadder

4.58.1.2 getCenter()

```
Vec2 getCenter (
            Rectangle r )
```

Return the geometrical middle point of the given rectangle.

Definition at line 22 of file rectangle.c.

```
22                               {
23     return (Vec2) {
24         r.position.x + r.size.x / 2,
25         r.position.y + r.size.y / 2
26     };
27 }
```

References Rectangle::position, Rectangle::size, Vec2::x, and Vec2::y.

4.58.1.3 isInside()

```
bool isInside (
            Rectangle inner,
            Rectangle outer )
```

Return true only if inner is fully inside of outer.

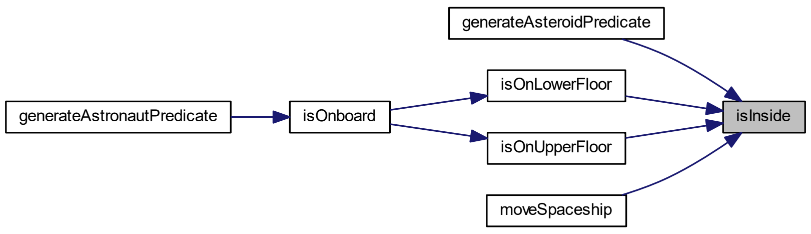
Definition at line 13 of file rectangle.c.

```
                                                    {
    return (
        outer.position.x <= inner.position.x &&
        inner.position.x + inner.size.x <= outer.position.x + outer.size.x &&
        outer.position.y <= inner.position.y &&
        inner.position.y + inner.size.y <= outer.position.y + outer.size.y
    );
}
```

References Rectangle::position, Rectangle::size, Vec2::x, and Vec2::y.

Referenced by generateAsteroidPredicate(), isOnLowerFloor(), isOnUpperFloor(), and moveSpaceship().

Here is the caller graph for this function:



4.58.1.4 translateRectangle()

```
Rectangle translateRectangle (
            Rectangle r,
            Vec2 translate )
```

Return a new rectangle shifted by vector translate.

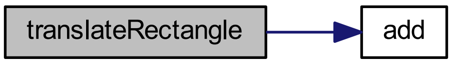
Definition at line 29 of file rectangle.c.

```
                                                               {
    return (Rectangle) {
        add(r.position, translate),
        r.size
    };
}
```

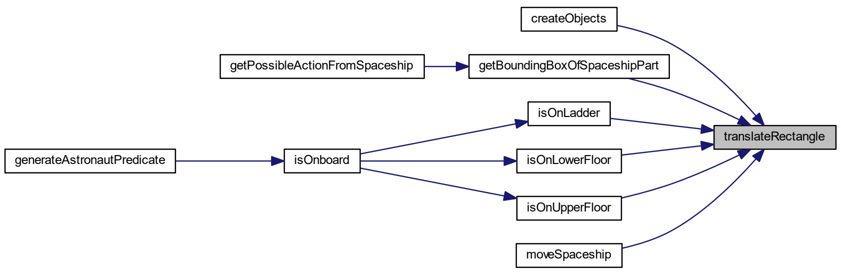
References add(), Rectangle::position, and Rectangle::size.

Referenced by createObjects(), getBoundingBoxOfSpaceshipPart(), isOnLadder(), isOnLowerFloor(), isOnUpper↩ Floor(), and moveSpaceship().

Here is the call graph for this function:



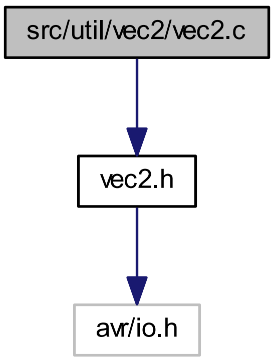
Here is the caller graph for this function:



4.59 src/util/vec2/vec2.c File Reference

```
#include "vec2.h"
```

Include dependency graph for vec2.c:



Functions

- Vec2 add (Vec2 v1, Vec2 v2)
- Vec2 substract (Vec2 v1, Vec2 v2)
- Vec2 clampVec2 (Vec2 v)

  Return new vector with ll components between -1 and 1 (inclusive)

Variables

- const Vec2 directions [ ]

4.59.1 Function Documentation

4.59.1.1 add()

```
Vec2 add (
            Vec2 v1,
            Vec2 v2 )
```

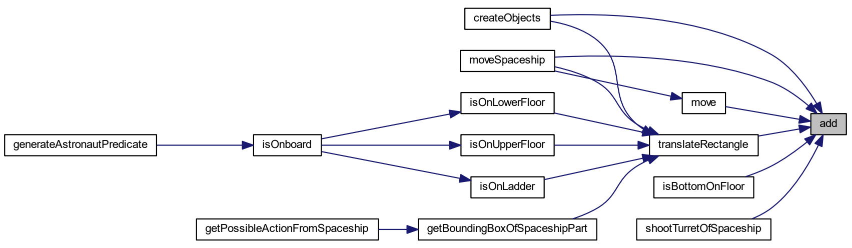
Definition at line 11 of file vec2.c.

```
11                          {
12     return (Vec2){v1.x + v2.x, v1.y + v2.y};
13 }
```

References Vec2::x, and Vec2::y.

Referenced by createObjects(), isBottomOnFloor(), move(), moveSpaceship(), shootTurretOfSpaceship(), and translateRectangle().

Here is the caller graph for this function:



4.59.1.2 clampVec2()

```
Vec2 clampVec2 (
            Vec2 v )
```

Return new vector with ll components between -1 and 1 (inclusive)

Definition at line 19 of file vec2.c.

```
                        {
    return (Vec2){
        v.x == 0 ? 0 : (v.x > 0 ? 1 : -1),
        v.y == 0 ? 0 : (v.y > 0 ? 1 : -1)
    };
}
```

References Vec2::x, and Vec2::y.

4.59.1.3 substract()

```
Vec2 substract (
            Vec2 v1,
            Vec2 v2 )
```

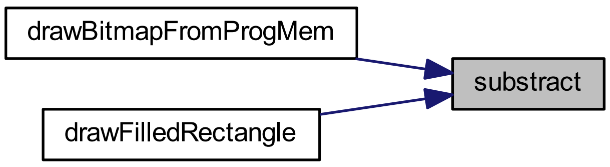
Definition at line 15 of file vec2.c.

```
                                {
    return (Vec2){v1.x - v2.x, v1.y - v2.y};
}
```

References Vec2::x, and Vec2::y.

Referenced by drawBitmapFromProgMem(), and drawFilledRectangle().

Here is the caller graph for this function:



4.59.2 Variable Documentation

4.59.2.1 directions

```
const Vec2 directions[]
```

Initial value:

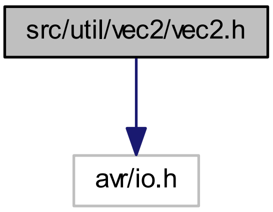
```
= {
    [north] = (Vec2){0, -1},
    [west] = (Vec2){-1, 0},
    [south] = (Vec2){0, 1},
    [east] = (Vec2){1, 0}
}
```

Definition at line 4 of file vec2.c.

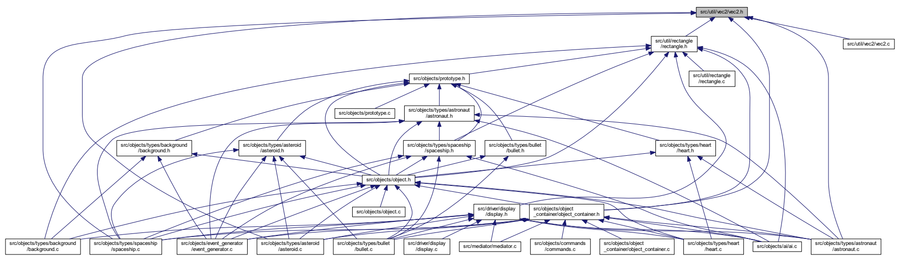
4.60 src/util/vec2/vec2.h File Reference

```
#include <avr/io.h>
```

Include dependency graph for vec2.h:



This graph shows which files directly or indirectly include this file:


src/util/vec2/vec2.h
src/util/rectangle/rectangle.h
src/util/vec2/vec2.c
src/util/rectangle/rectangle.c
src/objects/prototype.h
src/objects/prototype.c
src/objects/types/astronaut/astronaut.h
src/objects/types/background/background.h
src/objects/types/asteroid/asteroid.h
src/objects/types/spaceship/spaceship.h
src/objects/types/bullet/bullet.h
src/objects/types/heart/heart.h
src/objects/object.h
src/objects/object.c
src/driver/display/display.h
src/objects/object_container/object_container.h
src/objects/types/background/background.c
src/objects/types/spaceship/spaceship.c
src/objects/event_generator/event_generator.c
src/objects/types/asteroid/asteroid.c
src/objects/types/bullet/bullet.c
src/driver/display/display.c
src/mediator/mediator.c
src/objects/commands/commands.c
src/objects/object_container/object_container.c
src/objects/types/heart/heart.c
src/objects/ai/ai.c
src/objects/types/astronaut/astronaut.c

Data Structures

- struct Vec2

Enumerations

- enum Direction { north, west, south, east }

Functions

- Vec2 add (Vec2 v1, Vec2 v2)
- Vec2 substract (Vec2 v1, Vec2 v2)
- Vec2 clampVec2 (Vec2 vector)

  Return new vector with ll components between -1 and 1 (inclusive)

Variables

- const Vec2 directions [4]

4.60.1 Enumeration Type Documentation

4.60.1.1 Direction

```
enum Direction
```

Enumerator

north	
west	
south	
east	

Definition at line 12 of file vec2.h.

```
             {
    north, west, south, east
} Direction;
```

4.60.2 Function Documentation

4.60.2.1 add()

```
Vec2 add (
            Vec2 v1,
            Vec2 v2 )
```

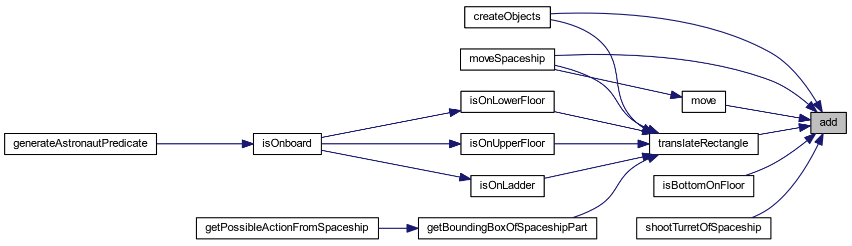
Definition at line 11 of file vec2.c.

```
                            {
    return (Vec2){v1.x + v2.x, v1.y + v2.y};
}
```

References Vec2::x, and Vec2::y.

Referenced by createObjects(), isBottomOnFloor(), move(), moveSpaceship(), shootTurretOfSpaceship(), and translateRectangle().

Here is the caller graph for this function:



4.60.2.2 clampVec2()

```
Vec2 clampVec2 (
              Vec2 vector )
```

Return new vector with ll components between -1 and 1 (inclusive)

Definition at line 19 of file vec2.c.

```
19                        {
20     return (Vec2){
21         v.x == 0 ? 0 : (v.x > 0 ? 1 : -1),
22         v.y == 0 ? 0 : (v.y > 0 ? 1 : -1)
23     };
24 }
```

References Vec2::x, and Vec2::y.

4.60.2.3 substract()

```
Vec2 substract (
              Vec2 v1,
              Vec2 v2 )
```

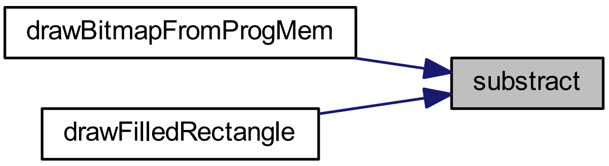
Definition at line 15 of file vec2.c.

```
15                                  {
16     return (Vec2){v1.x - v2.x, v1.y - v2.y};
17 }
```

References Vec2::x, and Vec2::y.

Referenced by drawBitmapFromProgMem(), and drawFilledRectangle().

Here is the caller graph for this function:



4.60.3 Variable Documentation

4.60.3.1 directions

```
const Vec2 directions[4]
```

The array directions can be indexed by a Direction and it contains vectors pointing into that direction.

Definition at line 18 of file vec2.h.