Table of Contents

Door Control API

API that configures whether the device is set as entrance/exit of the door, how the device will control the door, and the APB settings.

- BS2_GetDoor: Retrieves selected doors.
- BS2_GetAllDoor: Retrieves all doors.
- BS2_GetDoorStatus: Retrieves the status of selected doors.
- BS2_GetAllDoorStatus: Retrieves the status of all doors.
- BS2_SetDoor: Configures a door.
- BS2_SetDoorAlarm: Configures the alarm status of the door.
- BS2_RemoveDoor: Removes selected doors.
- BS2_RemoveAllDoor: Removes all doors.
- BS2_ReleaseDoor: Releases the lock/unlock flag of the door status. This initializes the priorities set to the door.
- BS2_LockDoor: Configures the priority of when the door gets locked. The lock priority must be higher than the unlock to lock the door.
- BS2_UnlockDoor: Configures the priority of when the door gets unlocked. The unlock priority must be higher than the lock to unlock the door.

Structure

BS2DoorRelay

```
typedef struct {
    uint32_t deviceID;
    uint8_t port;
    uint8_t reserved[3];
} BS2DoorRelay;
```

1. *deviceID*
Device ID.

2. *port*
Relay port number.

3. *reserved*
Reserved space.

BS2DoorSensor

```
typedef struct {
    uint32_t deviceID;
    uint8_t port;
    uint8_t switchType;
    uint8_t reserved[2];
```

```
} BS2DoorSensor ;
```

1. *deviceID*
Device ID.

2. *port*
Input port number.

3. *switchType*
Type of the switch.

Value	Description
0	Normally open
1	Normally closed

4. *reserved*
Reserved space.

BS2ExitButton

```
typedef struct {
    uint32_t deviceID;
    uint8_t port;
    uint8_t switchType;
    uint8_t reserved[2];
} BS2ExitButton ;
```

1. *deviceID*
Device ID.

2. *port*
Input port number.

3. *switchType*
Type of the switch.

Value	Description
0	Normally open
1	Normally closed

4. *reserved*
Reserved space.

BS2DoorStatus

```
typedef struct {
```

```
    uint32_t id;
    uint8_t opened;
    uint8_t unlocked;
    uint8_t heldOpened;
    uint8_t unlockFlags;
    uint8_t lockFlags;
    uint8_t alarmFlags;
    uint8_t reserved[2];
    uint32_t lastOpenTime;
} BS2DoorStatus;
```

1. *id*
Door ID.

2. *opened*
Determines whether the door is opened.

3. *unlocked*
Determines whether the door is unlocked.

4. *heldOpened*
Determines whether the door's status is held open.

5. *unlockFlags*
The priority of when the door gets unlocked, which will not operate if the priority is lower than the lock's priority. For example, if the door is locked with the operator priority, all users entry will not be allowed. The unlockFlags and lockFlags cannot have the same priority besides the default priority NONE.

Value	Description	Priority
0	None	Normal
1	Scheduled	High
4	Operator	Very high
2	Emergency	Highest

6. *lockFlags*
The priority of when the door gets locked, which will not operate if the priority is lower than the unlock's priority.

Value	Description	Priority
0	None	Normal
1	Scheduled	High
4	Operator	Very high
2	Emergency	Highest

7. *alarmFlags*
Status of the door alarm.

Value	Description
0	No alarm
1	Forced open
4	Held open
2	APB violation

8. *reserved*
Reserved space.

9. *lastOpenTime*
The last time of when the door was open.

BS2Door

```
typedef struct {
    uint32_t doorID;
    char name[BS2_MAX_DOOR_NAME_LEN];
    uint32_t entryDeviceID;
    uint32_t exitDeviceID;
    BS2DoorRelay relay;
    BS2DoorSensor sensor;
    BS2ExitButton button;
    uint32_t autoLockTimeout;
    uint32_t heldOpenTimeout;
    uint8_t instantLock;
    uint8_t unlockFlags;
    uint8_t lockFlags;
    uint8_t unconditionalLock;
    BS2Action forcedOpenAlarm[BS2_MAX_FORCED_OPEN_ALARM_ACTION];
    BS2Action heldOpenAlarm[BS2_MAX_HELD_OPEN_ALARM_ACTION];
    uint32_t dualAuthScheduleID;
    uint8_t dualAuthDevice;
    uint8_t dualAuthApprovalType;
    uint32_t dualAuthTimeout;
    uint8_t numDualAuthApprovalGroups;
    uint8_t reserved2[1];
    uint32_t dualAuthApprovalGroupID[BS2_MAX_DUAL_AUTH_APPROVAL_GROUP];
    BS2AntiPassbackZone apbZone;
} BS2Door;
```

1. *doorID*
Door ID.

2. *name*
Name of the door that will be displayed on the BioStar application.

3. *entryDeviceID*
Entry device ID.

4. *exitDeviceID*
Exit device ID.

5. *relay*
Door relay.

6. *sensor*
Sensor that detects the open/closed status of the door.

7. *button*
Exit button.

8. *autoLockTimeout*
Time for the door to lock after it has been opened. The unit is seconds.

9. *heldOpenTimeout*
Time for the door to be determined as held open. The unit is seconds.

10. *instantLock*
Decides whether to immediately lock the door when the sensor detects the door as closed.

11. *unlockFlags*
The priority of when the door gets unlocked, which will not operate if the priority is lower than the lock's priority. For example, if the door is locked with the operator priority, all users entry will not be allowed. The unlockFlags and lockFlags cannot have the same priority besides the default priority NONE.

Value	**Description**	**Priority**
0	None	Normal
1	Scheduled	High
4	Operator	Very high
2	Emergency	Highest

12. *lockFlags*
The priority of when the door gets locked, which will not operate if the priority is lower than the unlock's priority.

Value	**Description**	**Priority**
0	None	Normal
1	Scheduled	High
4	Operator	Very high
2	Emergency	Highest

13. *unconditionalLock*
Flag that decides whether to lock the door after autoLock timeout.

Value	**Description**
0	Locks the door only when the door is closed after the autoLockTimeout.

Value	Description
1	Locks the door regardless if the door is opened or closed.

14. *forcedOpenAlarm*
Alarm that gets triggered when the door is forced open, which can be configured up to 5 alarms.

15. *heldOpenAlarm*
Alarm that gets triggered when the door is held open, which can be configured up to 5 alarms.

16. *dualAuthScheduleID*
Schedule for the dual authentication. Set the value as 0 for disable, 1 for enable, or set a schedule ID.

17. *dualAuthDevice*
Decides which device should perform a dual authentication.

Value	Description
0	None
1	Only on entry device
2	Only on exit device
3	Both

18. *dualAuthApprovalType*
Decides whether to distinguish if the user belongs to an access group having authority when accessing the door .

Value	Description
0	None
1	Check the last user's authority

19. *dualAuthTimeout*
Interval between the first user's authentication and the second user's authentication. The unit is seconds.

20. *numDualAuthApprovalGroups*
Number of access groups having authority of dual authentication.

21. *reserved2*
Reserved space.

22. *dualAuthApprovalGroupID*
List of access groups having dual authentication authority, which can be configured up to 16 access groups.

23. *apbZone*
Configures Anti Passback on the door. The Anti Passback zone ID and door ID is equivalent. Refer Zone Control APIfor further information.