

NAME

`gsm_create`, `gsm_destroy`, `gsm_encode`, `gsm_decode` -- GSM 06.10 lossy sound compression

SYNOPSIS

```
#include "gsm.h"
```

```
gsm gsm_create();
```

```
void gsm_encode(handle, src, dst)
```

```
gsm handle;
```

```
gsm_signal src[160];
```

```
gsm_frame dst;
```

```
int gsm_decode(handle, src, dst)
```

```
gsm handle;
```

```
gsm_frame src;
```

```
gsm_signal dst[160];
```

```
void gsm_destroy(handle)
```

```
gsm handle;
```

DESCRIPTION

Gsm is an implementation of the final draft GSM 06.10 standard for full-rate speech transcoding.

`gsm_create()` initializes a gsm pass and returns a 'gsm' object which can be used as a handle in subsequent calls to `gsm_decode()`, `gsm_encode()` or `gsm_destroy()`.

`gsm_encode()` encodes an array of 160 13-bit samples (given as `gsm_signal`'s, signed integral values of at least 16 bits) into a `gsm_frame` of 33 bytes. (`gsm_frame` is a type defined as an array of 33 `gsm_bytes` in `gsm.h`.)

`gsm_decode()` decodes a `gsm_frame` into an array of 160 13-bit samples (given as `gsm_signals`), which sound rather like what you handed to `gsm_encode()` on the other side of the wire.

`gsm_destroy()` finishes a gsm pass and frees all storage associated with it.

Sample format

The following scaling is assumed for input to the algorithm:

0 1

11 12

S..v..v..v..v..v..v..v..v..v..v..v..v..v..v..*..*..*

Only the top 13 bits are used as a signed input value.

The output of `gsm_decode()` has the three lower bits set to zero.

RETURN VALUE

`gsm_create()` returns an opaque handle object of type `gsm`, or 0 on error. `gsm_decode()` returns -1 if the passed frame is invalid, else 0.

EXAMPLE

```
#include "gsm.h"

gsm handle;
gsm_frame buf;
gsm_signal sample[160];
int cc, soundfd;

play() { /* read compressed data from standard input, write to soundfd */

    if (!(handle = gsm_create())) error...
    while (cc = read(0, (char *)buf, sizeof buf)) {
        if (cc != sizeof buf) error...
        if (gsm_decode(handle, buf, sample) < 0) error...
        if (write(soundfd, sample, sizeof sample) != sizeof sample)
            error...
    }
    gsm_destroy(handle);
}

record() { /* read from soundfd, write compressed to standard output */

    if (!(handle = gsm_create())) error...
    while (cc = read(soundfd, sample, sizeof sample)) {
        if (cc != sizeof sample) error...
        gsm_encode(handle, sample, buf);
        if (write(1, (char *)buf, sizeof buf) != sizeof sample)
            error...
    }
    gsm_destroy(handle);
}
```

BUGS

Please direct bug reports to jutta@pobox.com and cabo@tzi.org.

SEE ALSO

`toast(1)`, `gsm_print(3)`, `gsm_explode(3)`, `gsm_option(3)`