Rockbuild

a build farm for open source projects

Björn Stenberg
bjorn@haxx.se

Background: Rockbox

- A firmware project for mp3 players
- 68 different players
- 4 binaries per player
- 5 cross-compilers
- 100+ committers

Building

How many targets do you build before you commit?

- Changes easily affect more targets than expected
- *Nobody* builds everything

Effects

- Popular targets work well
- Semi-popular get fixed regularly
- Unusual targets are mostly broken

Unacceptable!

Devs need to know if their change breaks a target build.

Obvious solution #1

Nightly builds

Obvious solution #1

Nightly builds

Problem: Not fine-grained enough.

Obvious solution #2

A script to easily compile all targets.

Obvious solution #2

A script to easily compile all targets.

Problem: 2-3 hour build time.

Obvious solution #3

Get a powerful server to build after commit.

Obvious solution #3

Get a powerful server to build after commit.

Problem: It still takes >1 hour.

Obvious solution #3

Get a powerful server to build after commit.

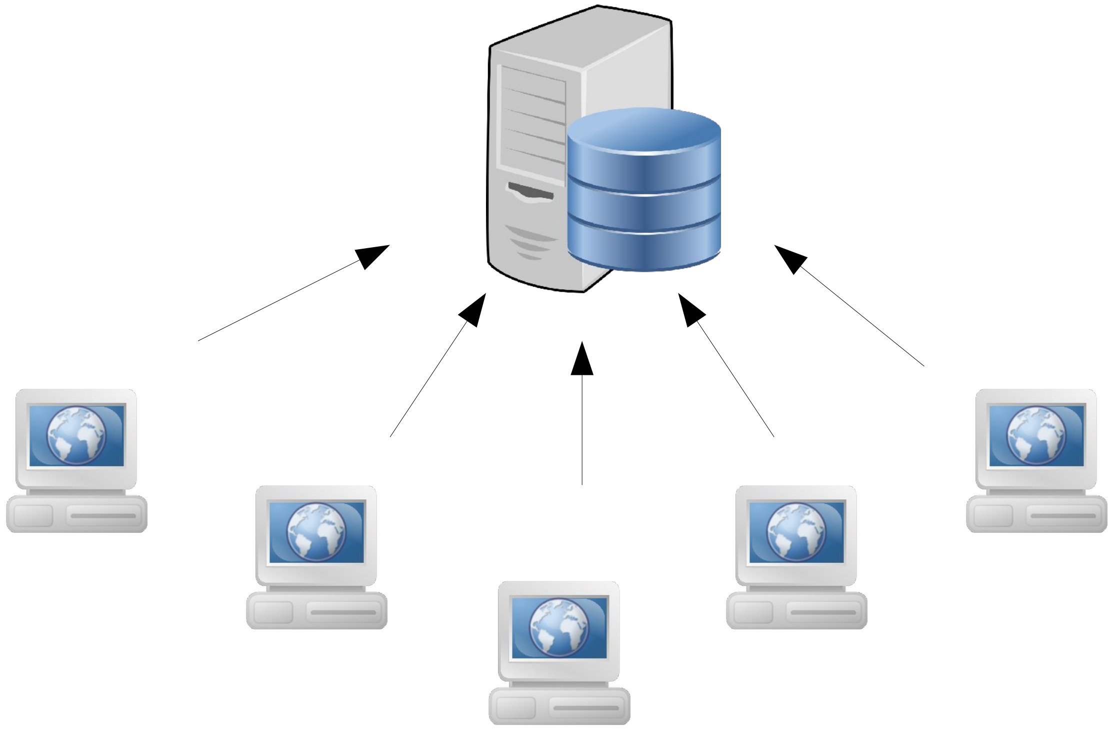
Problem: It still takes >1 hour.

Problem 2: Devs go away before build is done.

We need more power!

- One machine is not enough
- Buying more servers is expensive
- What to do?

Volunteer build farm



Volunteer machines

Challenges:

- Intermittent availability
- Firewalls & NATs
- Low tolerance for complexity
- Varying performance & capabilities
- Low bandwidth

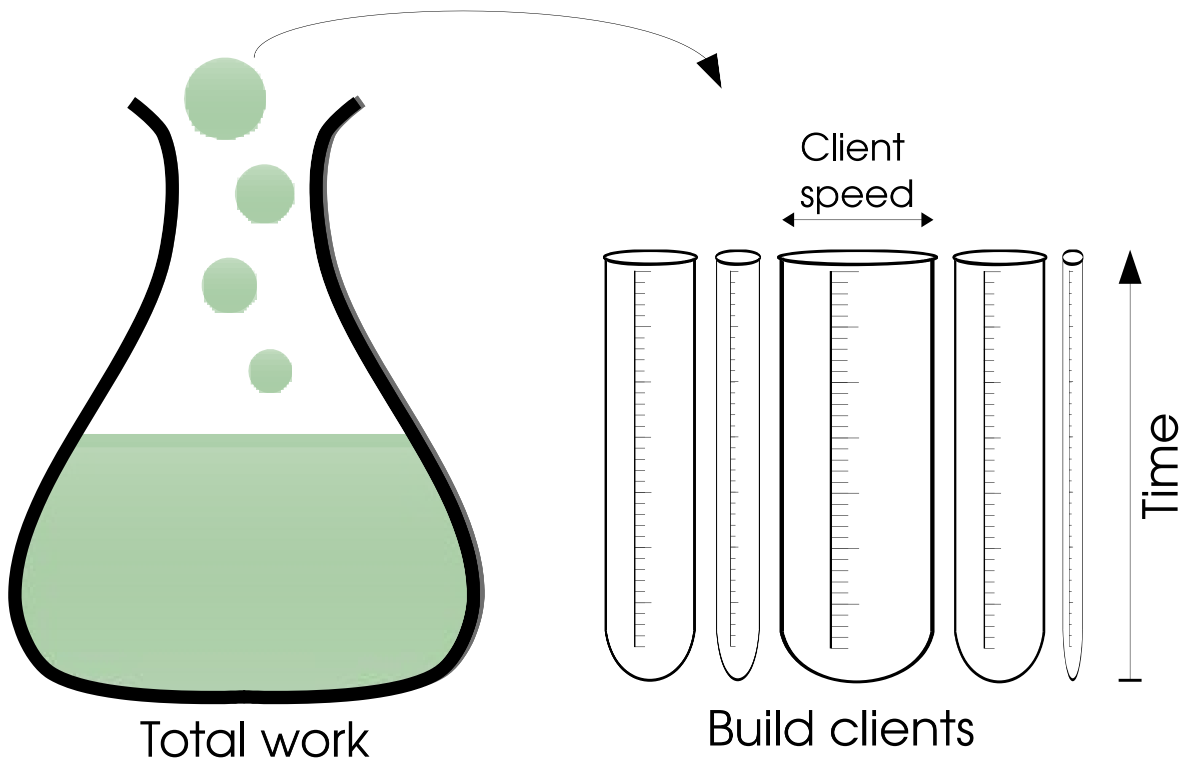
The simple build client

- No pre-registration
- No remote login
- Support any targets/compilers you like
- Come and go whenever you like
- You can be behind firewall & NAT
- Client is updated automatically
- Runs niced and selectable # of cores

The not-so-simple server

- Support a varying number of clients
- ...all running at different speeds
- ...supporting different subsets of targets
- ...and which may disappear at any time
- And finish quickly, please!

Distribution of work



Work planning

- Every target build has a “weight”
- Every client has an avg speed
- New clients are benchmarked using lightest builds
- Every client gets as much work as it can complete in time

Round time

- The weight of all work is known
- The speed of all clients is known
- weight / speed = time

Making a plan

- Work is in chunks, not liquid. There will be gaps.
- Iterate over clients, starting with the slowest. Assign as much work as they can complete in time.
- If all work doesn't fit, increase total round time and try again.

Reality ruins the plan

- Clients will disappear
- New clients will connect
- Clients won't perform as expected

= We need to be adaptive

Two build phases

- Phase 1: Cooperative phase
- Phase 2: Competitive phase

Cooperative phase

- Clients that finish their scheduled work are assigned unstarted builds from other clients.
- If a client disconnects, its work is marked as unassigned and picked up by other clients.
- If no client can build a target, it is dropped from the round.

Competitive phase

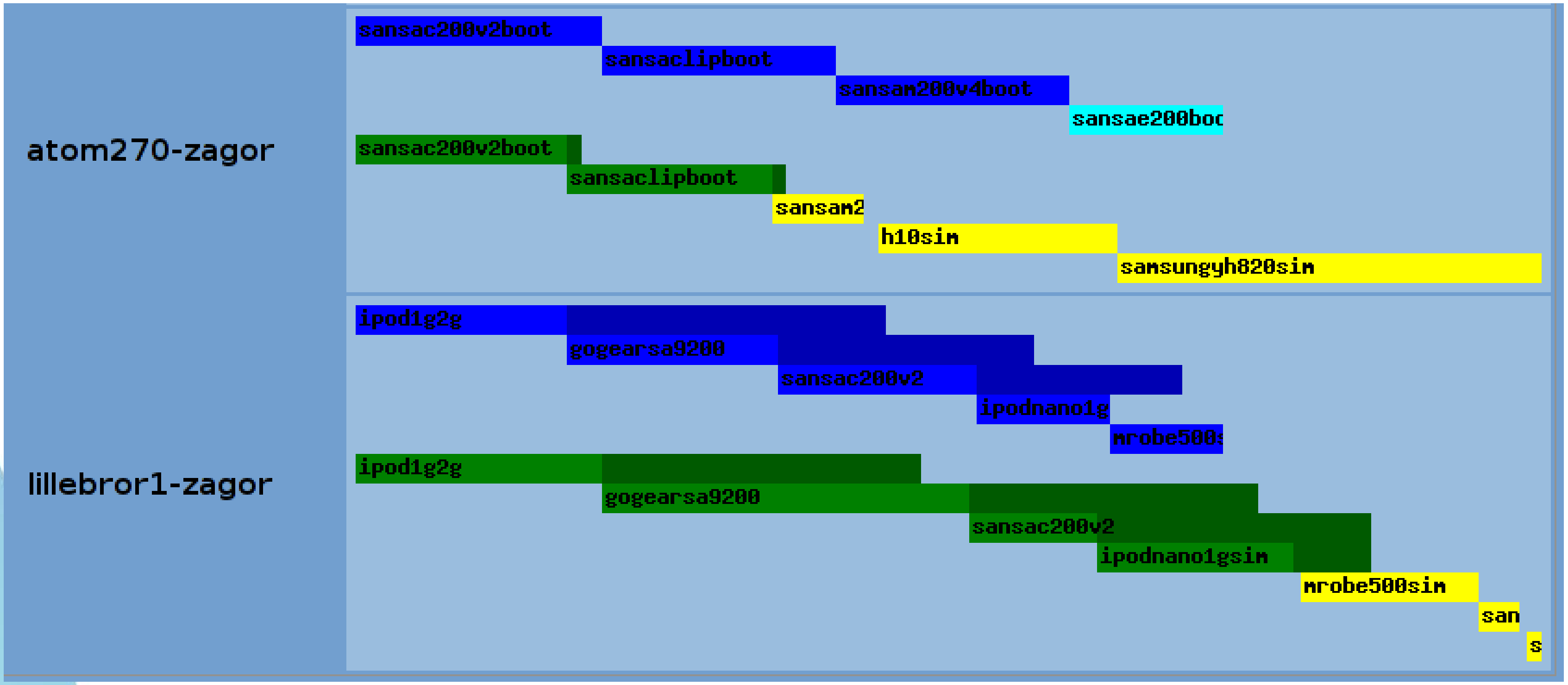
a.k.a "speculative"

- Clients start building targets other clients already build.
- This avoids getting delayed by an unexpectedly slow client.

Uploading

- When a client finishes a target, it forks an upload process and asks for next build job.
- This reduces upload speed as a performance factor. Only the last build in each round is affected.

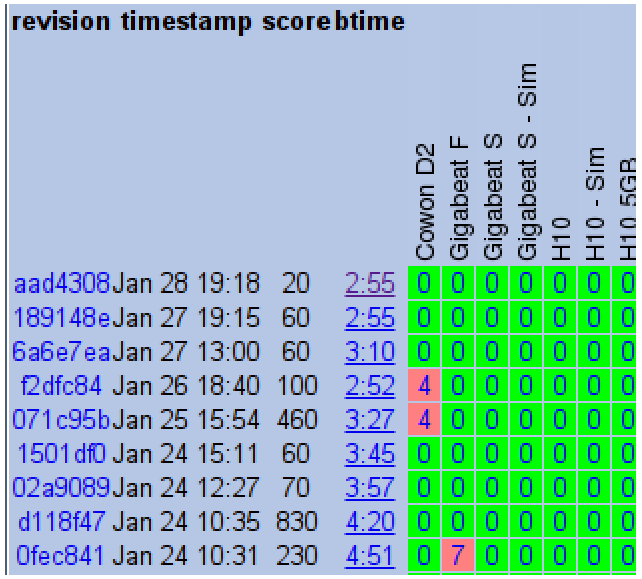
Example



atom270 = Atom N270 = 43 pts/sec
lillebror1 = Core i3 540 = 1089 pts/sec
(Factor 25 speed difference)

Build reports

HTML



revision	timestamp	score	btime	Cowon D2	Gigabeat F	Gigabeat S	Gigabeat S - Sim	H10	H10 - Sim	H10 5GB
aad4308	Jan 28 19:18	20	2:55	0	0	0	0	0	0	0
189148e	Jan 27 19:15	60	2:55	0	0	0	0	0	0	0
6a6e7ea	Jan 27 13:00	60	3:10	0	0	0	0	0	0	0
f2dfc84	Jan 26 18:40	100	2:52	4	0	0	0	0	0	0
071c95b	Jan 25 15:54	460	3:27	4	0	0	0	0	0	0
1501df0	Jan 24 15:11	60	3:45	0	0	0	0	0	0	0
02a9089	Jan 24 12:27	70	3:57	0	0	0	0	0	0	0
d118f47	Jan 24 10:35	830	4:20	0	0	0	0	0	0	0
0fec841	Jan 24 10:31	230	4:51	0	7	0	0	0	0	0

IRC

01:09 <CIA-138> Commit 97c1dc0 in rockbox by Michael Giacomelli: Enable logdiskf for all device targets, but not bootloaders or application.

01:12 <CIA-138> 97c1dc0 build result: 6 errors, 9 warnings (Michael Giacomelli committed)

Build logs

archosrecorder, revision 42a725f

Goto problem: 1

Built by hex-gevaerts

```
Build Command: ../tools/configure --target=archosrecorder --ram=2 --type=n && make zip
Using temporary directory /tmp
Platform set to archosrecorder
Memory size selected: 2 MB
[...]
OBJCOPY compressed.bin
make[1]: Leaving directory `/home/fg/rockbox/buildclient/rockbox/firmware/decompressor'
error: firmware image is 205278 bytes while max size is 204800!
make: *** [/home/fg/rockbox/buildclient/rockbox/build-archosrecorder/ajbrec.ajz] Error
255
Build Failure: No 'rockbox.zip' was produced.
```

Performance statistics

Build client stats, revision 7fda692

For these 213 builds, the following 36 build clients participated:

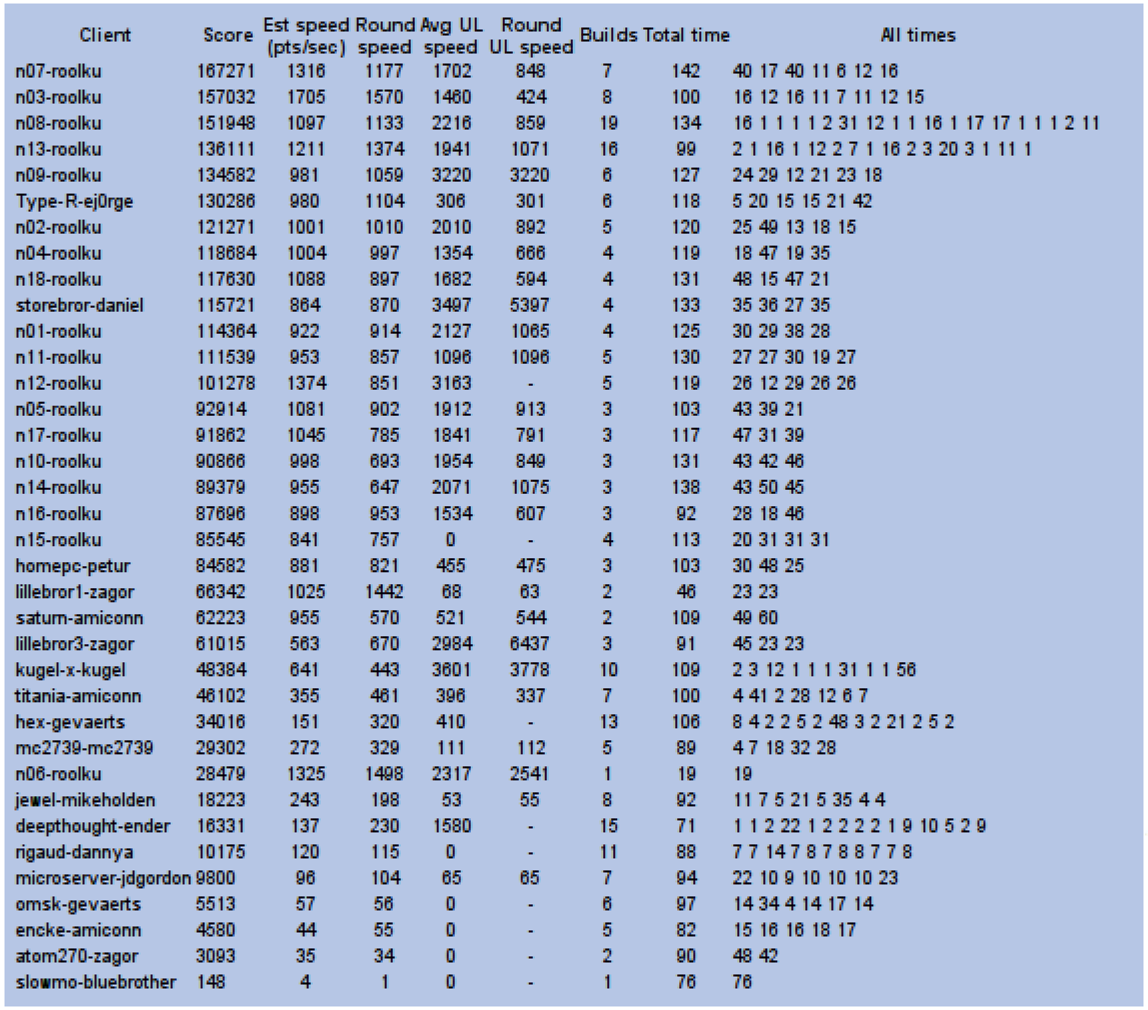
[[see next page]]

This build round took 151 seconds.

Total client speed was 25897 points/second, which in ideal conditions would complete the round in 105 seconds.

Effective round speed was 18174 points/second, making us **70% efficient**.

Performance statistics



Client	Score	Est speed (pts/sec)	Round speed	Avg UL speed	Round UL speed	Builds	Total time	All times
n07-roolku	167271	1316	1177	1702	848	7	142	40 17 40 11 6 12 16
n03-roolku	157032	1705	1570	1460	424	8	100	16 12 16 11 7 11 12 15
n08-roolku	151948	1097	1133	2216	859	19	134	16 1 1 1 1 2 31 12 1 1 16 1 17 17 1 1 1 2 11
n13-roolku	136111	1211	1374	1941	1071	16	99	2 1 16 1 12 2 7 1 16 2 3 20 3 1 11 1
n09-roolku	134582	981	1059	3220	3220	6	127	24 29 12 21 23 18
Type-R-ej0rge	130286	980	1104	306	301	6	118	5 20 15 15 21 42
n02-roolku	121271	1001	1010	2010	892	5	120	25 49 13 18 15
n04-roolku	118684	1004	997	1354	666	4	119	18 47 19 35
n18-roolku	117630	1088	897	1682	594	4	131	48 15 47 21
storebror-daniel	115721	864	870	3497	5397	4	133	35 36 27 35
n01-roolku	114364	922	914	2127	1065	4	125	30 29 38 28
n11-roolku	111539	953	857	1096	1096	5	130	27 27 30 19 27
n12-roolku	101278	1374	851	3163	-	5	119	26 12 29 26 26
n05-roolku	92914	1081	902	1912	913	3	103	43 39 21
n17-roolku	91862	1045	785	1841	791	3	117	47 31 39
n10-roolku	90866	998	693	1954	849	3	131	43 42 46
n14-roolku	89379	955	647	2071	1075	3	138	43 50 45
n16-roolku	87696	898	953	1534	607	3	92	28 18 46
n15-roolku	85545	841	757	0	-	4	113	20 31 31 31
homepc-petur	84582	881	821	455	475	3	103	30 48 25
lillebror1-zagor	66342	1025	1442	68	63	2	46	23 23
saturn-amiconn	62223	955	570	521	544	2	109	49 60
lillebror3-zagor	61015	563	670	2984	6437	3	91	45 23 23
kugel-x-kugel	48384	641	443	3601	3778	10	109	2 3 12 1 1 1 31 1 1 56
titania-amiconn	46102	355	461	396	337	7	100	4 41 2 28 12 6 7
hex-gevaerts	34016	151	320	410	-	13	106	8 4 2 2 5 2 48 3 2 21 2 5 2
mc2739-mc2739	29302	272	329	111	112	5	89	4 7 18 32 28
n06-roolku	28479	1325	1498	2317	2541	1	19	19
jewel-mikeholden	18223	243	198	53	55	8	92	11 7 5 21 5 35 4 4
deepthought-ender	16331	137	230	1580	-	15	71	1 1 2 22 1 2 2 2 2 1 9 10 5 2 9
rigaud-dannya	10175	120	115	0	-	11	88	7 7 14 7 8 7 8 8 7 7 8
microserver-jdgordon	9800	96	104	65	65	7	94	22 10 9 10 10 10 23
omsk-gevaerts	5513	57	56	0	-	6	97	14 34 4 14 17 14
encke-amiconn	4580	44	55	0	-	5	82	15 16 16 18 17
atom270-zagor	3093	35	34	0	-	2	90	48 42
slowmo-bluebrother	148	4	1	0	-	1	76	76

Bad clients

- Automatic temporary ban on client failure.
- Manual permanent ban

Security

- Unauthenticated build clients
- Auto-updated client code
- Not for building releases!

Who uses Rockbuild?

- Only Rockbox so far
- Recently made generic
- Not marketed

Other build systems

- distcc
  - distributes the building file-by-file
  - best suited for local clusters
- buildbot, hudson, continuum
  - static list of build clients
  - no client performance matching
- samba & postgresql custom farms
  - no commit builds

Thank you!

http://rockbuild.haxx.se

Björn Stenberg
bjorn@haxx.se